ISSN 0956-9979 SEPTEMBER 1998

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **NThing interesting?** It hardly seems like six months since the last *NT* comparative. Nineteen products, including a brand new name, line up for testing on p.14.

• **Fancy a cuppa?** The first Java virus was released in August. We have a technical analysis of it starting on p.11. A new in-the-wild macro virus with an unusual, data-stealing payload is analysed on p.9.

• **What's making the news?** This month's news stories range from the closure of the *NAI/Dr Solomon's* deal through a much-discussed network backdoor to another appearance of the Marburg virus in commercial software.

CONTENTS

EDITORIAL

As the Dust Settles Slowly in the West

After going to print last month, the media latched onto a new 'virus threat'. The moderator of the NTBugTraq mailing list, Russ Cooper, had highlighted the work of two Finnish security researchers who had uncovered buffer overflows in *Microsoft Outlook Express* and *Netscape Mail*.

As general computer security experts, their main concerns were (rightly) that such holes could be exploited to compromise *Windows* machines on a network. After all, email is not something that many organizations can afford to block at the firewall! Further, buffer overflows have been the bread and butter of Unix hackers for years, and whilst not simple to utilize in an exploit, there are 'cookbook' tutorials available on the web.

"Allusions to 'Good Times' were all but inevitable!"

Cooper wrote one of his occasional editorials on the situation. In it he very carefully pointed out that this had nothing to do with running executable attachments to messages. Unfortunately, the issue was clouded, in that this particular buffer overflow hangs off one of the MIME message headers associated with file attachments. Worse, in terms of adding confusion, the overflow was triggered (at least in *Outlook*) by simply clicking the icon that represents the list of attachments to a message in your in-box – you did not even have to open the message. Allusions to 'Good Times' were all but inevitable!

The general IT media lapped it up. Cooper was attending the Black Hat Briefings in Las Vegas and was inundated with calls. Reports started to appear suggesting that a virus in an attachment could be executed by using this buffer overflow. That is a theoretical possibility, but generally agreed to be a massive undertaking to write as an exploit – other, much simpler mechanisms are available.

And the focus on viruses seems odd. Why viruses? Because 'Good Times' – the quintessential email virus – had been mentioned?

Probably, which is unfortunate. The real threat here is quite different. This is, first and foremost, a security threat; not a code integrity issue. It seems very unlikely a virus author would go to the bother of trying to exploit this just to distribute a virus. On the other hand, someone targeting your company with an eye to filching technical secrets, staff or sales information or contacts may well find the effort pays off handsomely.

That may seem rather far-fetched. It certainly is not common, but if you do not know you have been targeted for such an attack, you most likely also do not know you have *not* been targeted. What is the probability of something like this happening? Low, yes – but negligible? I am not aware of an exploit utilizing this, but as with the perfect crime, technically better uses of it may well go undetected.

Back to *Outlook*. *Microsoft* released a patch for this bug (in Redmond-speak the 'long filename attachment vulnerability') and some related ones they discovered during code review for this fix. The fix affected several programs, including *Internet Explorer*. This news was broadly distributed with links to the *Microsoft* web site.

An enterprising fraudster took advantage of this to distribute a Trojan Horse. Emailed with spoofed headers suggesting that it came from iesupport@microsoft.com, the message claimed the attached file was a patch for *Internet Explorer*. The Trojan installed itself to load at *Windows* startup. When run subsequently, it tried, periodically, to send email messages from the host computer. The messages were randomly generated from sentence fragments – many vulgar or abusive. The recipients of these messages were the system administrator addresses at several Bulgarian ISPs.

This 'attack' was rather obvious to many who ran the Trojan. As the point was to inconvenience the target organizations, that probably did not bother its originator much. It is not a big step from this to a well-written, silent, document stealing Trojan. All your attacker needs then is a buffer overflow exploit to plant it via an otherwise innocuous email message…

NEWS

Aylesbury, Duck!

Late on Thursday 13 August 1998, the *Network Associates* (*NAI*) acquisition of *Dr Solomon's Software* was completed. On Friday morning, staff at the UK offices of *Dr Solomon's* were informed whether they had been made redundant or offered a position with *NAI*. Reports abounded – including those in the local Aylesbury paper, the Bucks Herald – of 60–80% of the UK staff being laid off.

Sources at *NAI* confirmed to *VB* that very few technical or support staff have been made redundant. At press time, some of the *Dr Solomon's* staff had not finalized their negotiations with *NAI*, so *NAI* was not prepared to present figures for job losses. Piecing together information from various sources, *VB* believes that around 200 people at Aylesbury were not offered positions at *NAI*. Approximately half of these were from production and stores (whose operations are being moved to an *NAI* facility in Holland) and administration. About a third of the lost jobs were in the only two technical departments to be trimmed – desktop QA and the *Toolkit 8* GUI team.

Most of the sales, support, consulting, development and virus lab staff have been retained. *NAI* confirmed to *VB* that it will maintain two anti-virus development centres – one in Aylesbury and one in Oregon. The detection engine team and groupware product development (*GroupShield*, *WebShield* and *NetShield for NetWare*) will be based in Aylesbury, with development of the desktop, *NT* and Macintosh products based in the US.

Staffing at *Dr Solomon's* German operation seems to have been largely unaffected by the takeover. Reports from the US offices have been much blacker, with suggestions that virtually all staff were effectively fired outright. Few have been prepared to talk at all, and none 'on the record'.

Following the deal's closure, the status of the companies' products and promised developments were revised. The eagerly-awaited *Toolkit 8* has now been shelved. The *Toolkit* will only be supported with driver updates. These are promised until the end of the third quarter of 1999.

NAI claims that the work of integrating the *AVTK* detection engine into *VirusScan* has progressed more quickly than expected. The major revision of *VirusScan* – version 4.0, originally scheduled for release in the first quarter of 1999 – is now scheduled to start shipping in October. A release for *NT* and server platforms will occur in November. A beta programme, open to all existing *NAI* and *Dr Solomon's* customers, will start this month.

NAI has also announced the retention of *Virex*, and that this product will replace *NAI's* current Macintosh offering. *Dr Solomon's* other products (*Audit*, *NetOctopus* and *Support Software*) have been retired ∎

Prevalence Table – July 1998

Virus	Type	Incidents	Reports
Autostart_9805	File (Mac)	77	16.8%
Laroux	Macro	50	10.9%
Cap	Macro	34	7.4%
AntiEXE	Boot	20	4.4%
Mental	Macro	18	3.9%
AntiCMOS	Boot	16	3.5%
Concept	Macro	16	3.5%
Form	Boot	15	3.3%
Win95/CIH	File	14	3.1%
CopyCap	Macro	13	2.8%
Extras	Macro	12	2.6%
Parity_Boot	Boot	9	2.0%
Empire.Monkey	Boot	8	1.7%
Mentes	Macro	7	1.5%
Ripper	Boot	7	1.5%
Wazzu	Macro	7	1.5%
DZT	Macro	6	1.3%
NYB	Boot	6	1.3%
Edwin	Boot	5	1.1%
Junkie	Multi-partite	5	1.1%
MDMA	Macro	5	1.1%
Stealth_Boot	Boot	5	1.1%
WelcomB	Boot	5	1.1%
Dodgy	Boot	4	0.9%
Npad	Macro	4	0.9%
Win95/Marburg	File	4	0.9%
Cascade	File	3	0.7%
DelCMOS	Boot	3	0.7%
HLL-CV	File	3	0.7%
Jumper.B	Boot	3	0.7%
ShowOff	Macro	3	0.7%
Stoned.Angelina	Boot	3	0.7%
V-Sign	Boot	3	0.7%
Others [1]		66	14.4%
Total		459	100%

[1] The Prevalence Table includes two reports each of: Appder, Breeder, ExeBug.G, Galicia.800, Hypervisor, Imposter, Johnny, Nottice, One_Half.3544, PolyPoster, Sack, Stoned.Standard, Swlabs, Tequila.2468 and Urkel.B; and single reports of: ABCD, AntiWin95, Barrotes, Beryllium, Bleah, Cartman, Czech, Dude, Eco, Enigma.730, Flip, Generic_Boot, Graveyard.479, Groov, Hybrid, Im_In.3141, Kompu, Lilith, Maniak, Muck, MultiAni, Munch, NiceDay, Ninja, Npox.1015, Quandary, Sampo, Schumann, Spirit, Stoned.Stonehenge, Temple, TPVO.3783, USTC.7680, Vacsina.1206, Werewolf.1208 and Wet.B.

VIRUS BULLETIN **©1998 Virus Bulletin Ltd, The Pentagon, Abingdon, Oxfordshire, OX14 3YP, England. Tel +44 1235 555139. /98/$0.00+2.50**

Up Yours!

The infamous anarchist/hacker/counterculture group Cult of the Dead Cow (cDc) garnered much press coverage recently with its release of *Back Orifice* (*BO*). Billed as a 'remote administration tool for *Windows 95*', *Back Orifice* 'gives its user more control of the remote *Windows* system than the person at the keyboard of that machine.'

In essence, *BO* provides some of the functionality of several commercial applications (such as *Symantec's PC Anywhere* and the remote administration elements of *SMS* from *Microsoft*) and a few similar freeware or shareware packages. The main differentiator for *BO*, should it be considered a serious contender in that market, would be price (free) and 'notoriety' – few developers choose the annual hacker conference DefCon as the launch venue for their products. [*Not if they hope to draw much mainstream support for it! Ed.*]

There are two components to *BO* – a server and a client. The server (default name BOSERVE.EXE) is *Windows 9x*-specific. Its function is to ensure that it is installed and configured to run at system startup. It provides network services so the client software can connect to a machine hosting it, and provides for remote access to many system and network APIs. The client was initially provided in Win32 console and GUI versions, but a subsequent port to Unix OSes (complete with source code) has been released.

BO provides the expected functionality of network administration tools, such as collecting system information, remote registry editing, file system browsing and the like. However, it also offers functions that seem less likely to have many legitimate uses. These include an HTTP server allowing file uploads and downloads, TCP and UDP port redirection, remote connection to most console applications via any network port, and network packet monitoring (allowing 'logging any plaintext passwords that pass').

The cDc publicity suggests that the *BO* server can be distributed 'attached' to another executable. On running and detecting this situation, *BO* reputedly detaches itself from its travelling companion, runs and installs itself and then launches the original program. This 'feature' certainly seems designed to facilitate Trojan-like distribution. In some limited testing *VB* is aware of, this feature has not been made to work, but one or two unconfirmed reports suggest this function can be made to work.

In light of *BO's* pedigree, many anti-virus developers have added detection of the server to their products. On balance, it has seemed more likely system administrators would wish to know that it is on their users' machines than not.

Perhaps to prove the cynics' point, several people have 'extended' *BO* in ways that suggest it will primarily be used for nefarious purposes. Some third-party 'wrappers' have appeared, designed to 'bind' BOSERVE.EXE to other executables, suggesting that *BO's* claimed native ability to achieve this does not work. The only use of these utilities is to turn some other program into a Trojan. At least one of these encrypts the *BO* installer with a randomly-generated key, making it a polymorphic Trojan. Those developers who have decided to detect *BO* had better be frequenting the right (or wrong) websites.

Apart from these external add-ons, *BO* is designed to support user-supplied functions. It does this via Back Orifice Unified Tool Transport plugins, or BUTTplugs. [*We are not making this up... Ed.*] These can perform any task their author desires. An early example broadcasts the *BO*-afflicted machine's IP address on IRC and another emails it to a pre-determined address (configurable to whatever is desired – in the case of a undesirable use, presumably an anonymous remailer).

To be a threat to your machine, the *BO* server must be executed on it. The simple precaution of not running any software other than that from the most trusted of sources is thus your best defence ∎

More-burg

Yet another commercial CD-ROM has shipped with files infected with Win95/Marburg. The new *MGM Interactive* game *Wargames PC* shipped in late July with the electronic registration program (\EREG\EREG32.EXE on the CD) infected. The game itself and the setup programs were unaffected and *MGM Interactive* is contacting people who register on-line, warning them of the infection and sending out a Marburg-specific disinfector. It is unclear as of this writing whether the game, or the infected pressing of it, is currently available outside North America, but reports there suggest that *MGM Interactive* has neither removed affected stock from the market, nor attached warnings to remaining stock! ∎

VB'98

Registrations are flowing in for this year's *VB* conference, to be held at the Hilton Park Hotel, Munich, Germany from Thursday 22 to Friday 23 October 1998. A welcome drinks reception is scheduled for Wednesday 21 October. Jimmy Kuo of *Network Associates* is presenting the keynote address, evocatively entitled 'Add Common Sense, Stir'.

The rest of the conference will again present corporate and technical streams. The former covers topics from devising a security and virus prevention strategy to ensuring software developers do not ship virus-infected material. The latter includes coverage of emerging Win32 virus attacks, Internet-borne threats and macro virus technicalities. The complete conference programme is available from our web site at http://www.virusbtn.com/VB98/.

VB'98 coincides with the second largest IT show in Europe. More information about *Systems'98* can be found at http://www.systems.de/ ∎

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 15 August 1998. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

- **C** Infects COM files
- **D** Infects DOS Boot Sector (logical sector 0 on disk)
- **E** Infects EXE files
- **L** Link virus
- **M** Infects Master Boot Sector (Track 0, Head 0, Sector 1)
- **N** Not memory-resident
- **P** Companion virus
- **R** Memory-resident after infection

Babylon.3081

CER: A polymorphic, appending, encrypted (in EXE files) 3081-byte virus containing the text 'Name: Babylon5 Cast Of Warriors (c) TechnoMag'. Files the virus has considered infecting have their time-stamps set to 44 seconds. Infected files have the word DEADh at offset 000Bh (COM) and at offset 0012h (EXE). Since EXE files are encrypted and polymorphic, the following template detects infected COM files only and may be used to detect the virus in memory.

```
Babylon.3081        B90D 00FC F3A4 B440 B909 0C33 D2E8 B8FB 7303 E9EC 00B8 0042
```

Havjiva.492

CR: An 492-byte appender containing the encrypted text '♦ Congratulations with come of the Spring !!! ♦Iehedarhed Havjiva'. It infects files starting with the byte E9h (NEAR JMP instruction).

```
Havjiva.492         B440 B9EC 01CD 21E8 1F00 B440 B105 BAA6 01CD 215A 59B8 0157
```

Hinder.380

CR: An appending, 380-byte virus containing the text 'Hinder II Ver 2.00 (c) 1995'. Infected files have the byte FEh at offset 0003h.

```
Hinder.380          2EC6 4703 FE5B B440 B97C 01BA 0002 E844 00B8 0042 B900 00BA
```

Hysteria.1784

CER: An appending, 1784-byte virus containing the texts 'This virus is created by Virus Generator On-Line' and '(c) 1998 Mad Daemon / maddaemon@hysteria.sk'.

```
Hysteria.1784       B9F8 0631 D2B4 40E8 EDFA 3DF8 0675 2680 3EF8 064D 740A B907
```

IVP.858

CN: An encrypted, appending, 858-byte, fast, direct infector containing the texts 'You are a looser !!!', 'Infilsator V0,02 BRubellerThis is an Infeckt File, Infilsator act', 'This is INFILSATOR', '.....and I Infill Yoour Diskssss', 'FORMAT.COM Formatiert die Festplatte', 'EINGABE für Formatieren!', 'Alle Daten werden gelöscht! Weiter[J/N] ? J' and '*.com'. Infected files have 4943h ('CI') at offset 0003h.

```
IVP.858             0133 C0B9 0E03 5852 2E30 2433 D2BA 0600 4683 C208 E2F2 5AC3
```

Jpage.821

CN: An appending, 821-byte, fast direct infector containing the texts 'Jimmy Page Virus | pt.1 of the LeD zEpPeLiN Virus Family', 'Made in the USA', '????????COM', 'Jimmy Page Virus (c) 1995 | Written By: Wicked Rage', '*.com' and 'Led Zeppelin | Jimmy Page | Led Zeppelin'.

```
Jpage.821           3E88 A676 03B9 3503 BA02 01B4 40CD 21B8 0042 33C9 33D2 CD21
```

KeyKiller.665

CR: A stealth, appending, 665-byte virus which intercepts and plays tricks with keystrokes. It contains the encrypted text 'Key Killer, (c) 1995 by Mega Devil in AZORES!!! - Portugal.'. Infected files have their time-stamps set to 62 seconds.

```
KeyKiller.665       50B4 40B9 9902 0E1F BA03 01CD 2133 C933 D2B8 0042 CD21 2EC6
```

Mandragore.664

CR: A stealth, 664-byte virus containing the texts 'Mandragore', '[Mdrg v5]', 'BEER and TEQUILA forever !', 'Error 8869: processor drunk 8*) !' and 'Eddy iz still alive somwhere in time .....'. Infected files have their time-stamps set to 2 seconds.

```
Mandragore.664      B43F B903 00BA 5E00 FEC4 5050 CD21 58B9 9502 33D2 CD21 B800
```

Padania.2547

CER: A polymorphic, 2547-byte appender containing the texts 'Sailor_Pluto.b', '-b0z0/iKx-', '[SMPE 0.2]','PADANIA', and 'TBAVF-SCMSFINACO'. No simple template can be provided for this virus.

Rat.848

EN: A appending, 848-byte, slow, direct infector containing the texts 'Help 11-02-97' and '♥Charlene Samuel♥DD#I♥CS###'. Infected files have the text 'RAT' at the end of their code.

```
Rat.848             B950 03E8 FAFE E801 FFB4 3FB9 1800 BA89 02CD 21E8 EFFE B900
```

RPME

CN: Two overwriting, polymorphic, fast, direct infectors. The code written to infected programs is a large polymorphic procedure which builds the infector on the fly in memory. After the infecting part is constructed it contains the texts 'RPME v.02 by RedArc' and '*.com'. The .A variant is 4998 bytes and the .B variant size varies around 4200 bytes. It is impossible to use a simple template to detect either.

RPME.C

EN: An overwriting, polymorphic, 5495-byte, fast, direct infector. It contains the texts 'RPME v.01 by RedArc' and '*.com'. No simple template can be provided for this virus.

RPME.Companion

EN: Two polymorphic, direct infecting, companion viruses. They build the infection procedure on the fly. After it is constructed it contains the texts 'Wandering Byte' and 'RPME v.02 by RedArc'. The .A variant changes the host's extension to .OVL and writes itself to the EXE file. The .B variant is a minor modification that changes the host's extension to .DAT. No simple template can be provided for either.

Soldier.1480

CER: A stealth, appending, 1480-byte virus containing the texts 'Soldier BOB - (c)jan-94 by A:N:O:I', 'Programmed by Macaroni Ted', 'Soldier BOB - Made in Sweden.', '*.com', '*.exe' and ''output.'. Infected files have their time-stamps set to 28 seconds and have the word 4941h ('AI') at offset 0003h (COM) and at offset 0012h (EXE).

```
Soldier.1480        B84E FFCD 213D 494F 7418 1E06 B411 CD21 80C5 042E 882E 7501
```

Smile.5504

MCER: A multi-partite, 5504-byte virus which infects MBRs and prepends itself to executable files. Similar to an earlier variant (infected MBRs are identical), it is often detected as Smile.4320.A (or Yesmile.4320.A). The virus is named for its payload, producing a laughing sound through the speaker.

```
Smile.5504 (MBRs)   0600 A313 04B1 06D3 E08E C0B8 B902 0B90 00BA 8000 CD13 BB7C
Smile.5504 (files)  720F B800 57E8 C0FE B801 5780 C91F E8B7 FEC3 3FDC CC66
```

Trip.1952

EN: A direct, 1952-byte infector infecting three files at a time. It contains the texts 'Hardware detection error: CPU bad or missing.', 'Exception #0D: Attempt to write to write protected CPU register.', 'DMA failure: access is too direct.', 'Access violation: application terminated', 'DPMI error #013: HMA is too high, need line A21.', 'Divide overblow (or ...flow, do you know?).', 'By the way, do you know what job is blow?', 'Incorrect DOS perversion.', 'Fucked file corrupt.', 'Virus warning: This file is possibly infected by Tripper!', 'Incorrect user version, upgrade, please.', 'You bastard! Put the diskette in right now!', 'Tripper message: $One more file succesfully infected by Tripper.', 'This program must be run under Win32.', 'Abnormal program termination: consult with her psychiatrist.', 'fuck: jmp far 0ffffh:0000h', 'Oracle(R) proclaims: Enabling Information Age Through the Network Computing.', 'Hitler kaput!', 'Lozinsky - woodpecker, AIDSTest - <zensored!> :)', ' Tripper(R) Version 1.0 for Intel-8086(R) (C)Copyright Golem 01/08/98(1998)', 'Shit! A fucking error occured!', 'Write protection will not save you!' and 'vir???*.exe'. Infected files have the string: '♂Tripper♀' at the end of code.

```
Trip.1952           A1B8 07A3 8C07 B9A0 07B4 40CD 2172 4233 C933 D2B8 0042 CD21
```

Trivial.628

CEN: An overwriting, 628-byte, direct fast infector containing the texts 'C:\windows\command', 'c:\command.com', 'c:\windows\system', 'c:\windows', '*.bat', 'DIE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!' and 'I AM ERIN-APLHA,COUSIN OF ERIN-OMEGA, NOW YOU MUST LEARN THE HARD WAY TO BOW DOWN TO THE ALMIGHTY VIRUS'.

```
Trivial.628         B440 B974 02BA 0001 CD21 B43E CD21 B44F EBDC B43B BA9E 02CD
```

Tsunami.3001

CER: An encrypted, appending, 3001-byte virus containing the texts '86ANOTAVCOTB', 'Tsunami.PNG -1nternal', 'ANTI-VIR.DAT' and 'CHKLIST.MS'. The virus avoids infecting files with names containing digits. The following template can be used to detect the virus in memory only.

```
Tsunami.3001        8B1E 0300 B996 0B33 D2E8 B3FF 1FC3 3836 414E 4F54 4156 434F
```

Tsunami.3011

CER: An encrypted, appending, 3011-byte virus containing the texts 'Tsunami.PNG -1nternal', 'ANTI-VIR.DAT', '86ANOTAVCOTB' and 'CHKLIST.MS'. The virus infects only files with names ending with 'VX'. The following template can be used to detect the virus in memory only.

```
Tsunami.3011        8B1E 0300 B9A0 0B33 D2E8 B3FF 1FC3 3836 414E 4F54 4156 434F
```

Twin.351

ER: A companion, 351-byte virus containing the texts 'COMEXE' and 'COM'.

```
Twin.351            B810 FFCD 213C 0775 07E8 2300 B44C CD21 B821 35CD 212E 891E
```

Vietnow.577

CN: An encrypted, appending, 577-byte, fast, direct infector. It contains the texts '*.com' and '———————————— VietNow ——— By Arsonic[CB] ———————————————— Fear is Your Only ——— God! ——————————————'.

```
Vietnow.577         5F01 8AA6 5F01 32A6 5D01 80EC 0286 C4FE C0FE C0F6 D0AB E2DA
```

VB.529

ER: An appending, 529-byte virus. Infected files have the word 4256h ('VB') at the end of code.

```
VB.529              B8DD 4BCD 213D 4BDD 7469 1EB4 4ABB FFFF CD21 83EB 2490 B44A
```

Xres.395

ER: An appending, 395-byte virus residing in the Interrupt Vector Table. Infected files have the byte 6Bh ('k') at offset 001Ah.

```
Xres.395            B98B 0190 FCF3 A48E D8BA 3802 B821 25CD 2107 1F58 2EFF 2E4B
```

Yez.1155

CER: An appending, 1155-byte virus containing the texts 'COMMAND COM' and 'YeZ =PhVx= Article virus (demonstration purposes only)...'.

```
Yez.1155            33D2 B983 04E8 3500 3BC1 7404 F9EB 0790 32C0 E82E 00F8 C3B8
```

Yunk.525

CN: An encrypted, appending, 525-byte, direct infector infecting one file at a time. It contains the texts '*.*' and '*.COM'. Infected files have the word 3412h at the end of code. Due to the encryption it is impossible to select a reliable template for this virus.

Zlodic.999

CN: An encrypted, appending, 999-byte, direct, fast infector containing the texts '-*Zlodic.999*-MIEM-*.CoM' and '*.eXe'. Infected files have the byte 90h at offset 0003h.

```
Zlodic.999          013E 8ABA 5301 32F8 32FC 3E88 BA53 0183 C602 3BF1 7702 EBDD
```

VIRUS ANALYSIS 1

Taking the Libertine

Eugene Kaspersky
Kaspersky Lab, Russia

Virus attacks on *Microsoft Windows* continue. More and more *Windows*-specific viruses are appearing and some, like Marburg and CIH (see *VB*, August 1998, p.3 and p.8 respectively) have, unfortunately, become well-known. These viruses use new technologies to infect *Windows* executables, test new methods of staying memory-resident and yet do not forget good old DOS!

Win95/Libertine is a multi-platform virus. It infects both DOS COM and Win32 Portable Executable (PE) files, stays resident in *Windows* as a task, and periodically searches for files and infects them. Additional features include an amusing payload, anti-anti-virus capabilities, and polymorphism in infected *Windows* executables.

Despite being written in Assembler, Libertine is surprisingly large – about 31 KB (31,672 bytes, to be precise). Fortunately, 16 KB of that is a JPEG image, dropped in the payload. Furthermore, a 7 KB block is occupied by tables, PE relocation and import segments reserved for the virus' data buffers; a 3 KB block forms the polymorphic entry code; and only about 5 KB of the total virus contains subroutines which must be analysed to disclose all of Libertine's secrets.

Win95/Libertine

This virus is a multi-platform, polymorphic infector which attacks COM and PE executables. It was named from a text string in its code:

```
[Win32.Libertine v1.07b]
Copyright 1998-xxxx by <NeverLoved>
```

The virus' author clearly did not understand that infecting PEs is insufficient justification to label a virus as Win32.

Apart from its two infective forms, Libertine is found as a 31,672 byte PE dropper. Due to bugs, infected COM and EXE files cannot run under *NT* and terminate with standard *NT* or DrWatson error messages. Thus, the virus is properly named Win95/Libertine.31672. The Libertine dropper runs under *NT* with no problems, but its children cannot complete the circle of infection, so it is not a true Win32 virus.

While infecting both PE and COM files, Libertine writes its entire 32 KB of code to the end of the host, modifying file headers to pass control to the virus. The addresses of the entry routines are different in the three possible cases.

When Libertine takes control in infected PE and COM programs, it searches for the dropper (C:\MYLENE.EXE), executes it and returns control to the host. If the dropper does not exist the virus first creates one then executes it. These dropper creation and activation routines are short and simple. All viral functions, such as infection and payload, are performed by the dropper.

Running the Dropper

When the dropper is executed, it first performs some tricks to hide its presence in the system. In order to prevent access violation messages, it uses the KERNEL32 SetErrorMode function to disable General Protection Fault error messages. *Windows* will just continue executing the application without reporting protection violations.

The virus then checks which system is running, and in the case of *Windows 9x* patches the undocumented system process database, setting two flags – NukeProcess and ServiceProcess. These flags make the process invisible to the Ctrl-Alt-Del task list, and prevent it being terminated when the user logs off. To enable these flags, the virus then re-executes its dropper with the argument 'sexy'. Successfully executed under a patched system environment, the virus sets about its other tasks.

Anti-anti-virus Routine

Libertine targets one anti-virus program – the *AVP Inspector* (*AVPI*) integrity checker (CRC scanner). It uses the registry to obtain the path to *AVPI*, from which it opens AVPICHCK.DLL and scans it for a specific code sequence. If the code is found it is replaced with NOPs. Depending on the version of *AVPI*, this 'patch' causes it either not to detect changes, or immediately to display a statistics dialog.

To prevent duplicate patching, it stores the string 'kcah' ('hack' backwards) in the file header at offset 0030h. If *AVPI* is running, Libertine obtains the thread ID of the AVPI32 window, opens it, then terminates the process.

Infection

When the infection routine gains control, Libertine scans all the files in all the subdirectory trees on all installed hard drives, starting from drive C. If there is a non-fixed disk in the sequence (CD-ROM, remote, or other) the virus terminates itself. When a file is found, Libertine checks for a COM or EXE file extension. This virus only infects such files and searches for the next suitable file or subdirectory entry. With seven out of eight potential hosts (depending on the system timer), it continues the infection process.

Initially, COM and EXE hosts are separated by comparing their first two bytes to the magic 'MZ' stamp. In EXE files, Libertine then checks for Portable Executable, and its own stamps ('PE' at the top of the header and 'I_M!' in the PE checksum field). While adding its code to the end of PE

files, the virus creates a new section named _Mylene_, modifying the Entry Point and some other necessary header fields. It then runs the polymorphic engine, encrypting a copy of itself and writing this to the end of the file in the newly-created section.

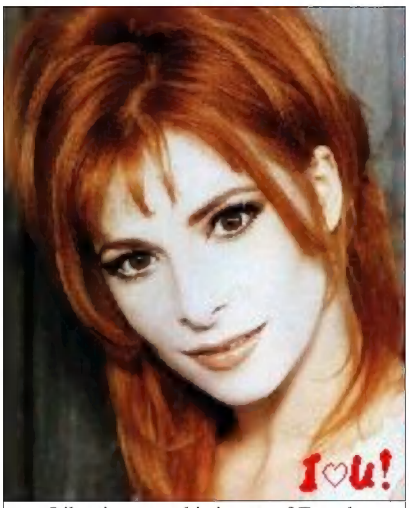
With COM files, Libertine also writes itself to the end of the file, then converts the file format to EXE by adding an EXE header. This means that it does not infect COM files twice because of its internal file format check. Nor does it infect COM files with the 'ENUN' text string at the end. This indicates that the files are protected by a CRC self-check (see 'Addendum', *VB*, December 1997, p.9), so the virus avoids them. Finally, Libertine does not infect COM files smaller than 2 KB or those larger than 60 KB.

Infected COM Files

While infecting COM files, Libertine converts them to EXE format with the appropriate headers – MZ magic stamps at the top and all the necessary EXE header fields. Careful investigation of these headers shows that several fields have values that are not found in ordinary programs.

Firstly, the HeaderSize (Paragraphs in Header) field is set to zero, but there must be at least two bytes – 'MZ'. Another suspicious field is 'CS at entry' with a value of -10h (FFF0h). Using both of these causes DOS to load the file as a standard COM file. The bytes from 0000h to 00FFh are occupied by the Program Segment Prefix, bytes 0100h to the end of the allocated memory are for the file image, and the very first byte is placed at offset 0100h. In COM files the file image cannot overlap a segment (64 KB), but in EXE files the loadable image can be of any size that fills one block of free memory.

This trick enables the virus to infect COM files of any size. After conversion they do not lose their functionality, even if their size grows over the normal 64 KB limit on COM files. Recall Libertine's size – without converting a COM to the EXE format, the virus would not be able to infect COM hosts larger than 32 KB. The conversion also provides an easy method to return control to the host program – Libertine simply restores the first 4Fh bytes of the file header and jumps there. The code is loaded into the memory from the very first bytes ('MZ') to the very last, and control is passed to the entry point in the EXE header. In Libertine's case, the entry code is placed very near the top of the file – at 0020h.



Libertine uses this image of French singer Mylène Farmer in its wallpaper-changing payload and the title of an early song of hers for its name.

By using 32-bit i386 instructions, this routine gets the offset of the main virus code (stored at 001Ch in the header), converts it to 16-bit segments and jumps there. The entry routine also hooks Int 24h to prevent standard error messages appearing while writing to write-protected volumes.

When the main virus code receives control it checks whether *Windows* is running. If not, or if the version is less than 4.0, the virus restores the bytes from the original host program at its top and returns control there. If *Windows* is active, Libertine creates the C:\MYLENE.EXE file, writes the dropper there, closes the file and executes it. Then the dropper takes control and returns to infection level.

Infected PE Files

Executing an infected PE file hands control to Libertine's polymorphic routine, which decrypts the virus code and jumps to the main routine. This is different from the COM file code. It is a 32-bit program that operates with *Windows* memory and resources. However, the objective remains the same – it creates and executes the dropper.

To access the *Windows* functions it needs, the virus obtains their addresses by scanning the *Windows* kernel. Libertine checks the environment (*Windows 9x* or *NT*) and uses the correct offsets in both cases, parsing KERNEL32 imports searching for CreateFileA, WriteFile, CloseHandle and WinExec functions. The virus' infection procedure is quite simple and does not need more than these four entries. A bug in this code prevents Libertine spreading under *NT*.

The file C:\MYLENE.EXE is created, the dropper's code written there, the file closed and executed. The dropper takes control, and this PE branch of the virus algorithm returns to the root.

Payload

Before calling anti-anti-virus and infection routines the dropper form calls its payload. This is executed with a one in eight probability, depending on the system timer, and changes the desktop background (wallpaper). Libertine writes an image of Mylène Farmer (a popular French singer) to the file C:\MYLENE.BMP, converts it to BMP format and sets it as the system wallpaper.

To achieve this Libertine examines the system registry key HKLM\SOFTWARE\Microsoft\Shared Tools\Graphics Filters\Import\JPEG. If such a filter is registered, the corresponding library is loaded and its ImportGr routine is called. Use of these routines reduces Libertine's size significantly, with the 16 KB JPEG producing a 160 KB BMP file. *Windows 9x* only accepts BMP files as wallpaper images, so the virus has to convert the JPEG it carries to BMP. The resulting image is saved to the same file and is

then registered as the *Windows* desktop wallpaper. Despite having everything necessary on my test computer, Libertine failed to change my wallpaper.

Direct Action but Memory-resident?

Before infecting each file, Libertine calls the standard *Windows* API Sleep function and delays activity for three seconds. This is the most important block of virus code. This one call qualifies the virus as 'memory-resident'. The virus sleeps for three seconds between file infections. If there were 1000 COM and EXE files on a PC it would stay in memory for 50 minutes (3000 seconds).

Although it stays in system memory for a long time, and other applications may be run while the virus infection routine is active, Libertine is a direct action infector. This is how typical non-resident viruses work – no events are hooked, it searches for files in subdirectory trees and infects them, then exits.

Closing Comments

Virus writers are still creating *Windows* VxDs which hook IFS APIs, much as DOS TSRs hook Int 21h. It is still the most popular way to create memory-resident *Windows* viruses nowadays. Maybe Libertine is the first virus with a semi-resident feature, and in future we will see more such viruses – after all, it is much easier to debug a *Windows* application than a *Windows* VxD or *NT* driver.

Libertine

Aliases: Win95/Libertine.31672.

Type: Parasitic COM and PE infector. Direct action virus, but is active in Win32 system memory to the end of its search-and-infect phase.

Self-recognition:

In PEs: Compares the CRC field in the PE header with 'I_M!'.

In COMs: Converts them to EXE format and does not infect DOS EXE files.

Hex Pattern:

PEs: The virus is polymorphic, so there is no hex pattern to detect it. Infected files will have a section named _Mylene_.

COMs, dropper and memory:

```
6A03 E886 1000 00C6 05AF 7040
0001 8A44 2403 3CBF 752E E830
1000 008B C864 67A1 1800 83E8
1033 C1A3 9370 4000 E814 1000
```

Trigger: Installs new Windows wallpaper.

Removal: Under clean system conditions identify and replace infected files. Delete the C:\MYLENE.EXE file.

VIRUS ANALYSIS 2

Lock Up your Data!

Gábor Szappanos
Computer and Automation Institute, Hungary

As far as global tendencies in computer technology are concerned, Hungary has always been a couple of years behind. We are not proud that this technology lag is possibly shortest in the creation of computer viruses. So the great surprise is not that the first home-brew macro virus has appeared, but that it is not a simple Concept rewrite. In fact, WM/Mentes is quite a sophisticated specimen with an interesting and unique payload.

In a Nutshell

This virus has appeared in the wild in several places in Hungary, and has been reported outside the country too. It consists of ten execute-only macros: Killer, AutoClose, FileSave, FileSaveAs, ToolsMacro, AutoExec, DocClose, ListMacros, FileOpen, and AutoOpen – a total effective length of 3820 bytes. The same set of macros is used in the infected global template as in infected documents.

The Killer macro is the largest of all. It contains procedures for infecting documents (MENTES and TERJED – Hungarian for Save and Spread respectively) and for the eventual removal of Mentes' macros (MAIN). The other macros call these procedures and can be divided into three groups. The first group – FileSave, FileSaveAs, DocClose, FileOpen and AutoOpen – is responsible for infection. The AutoOpen and FileOpen macros have additional responsibility for the virus' partial self-removal. Members of the second group – ToolsMacro and ListMacros – provide the virus' limited stealth capabilities, while the AutoClose macro activates the payload. The AutoExec macro does not belong to any of the groups. It simply enables automacro execution.

Infection

The global template is infected when a Mentes-bearing document is opened (via AutoOpen or FileOpen) or closed (DocClose), or when an infected document is saved (FileSave, FileSaveAs). From then on, any document that is opened, closed or saved is infected. The virus uses the Killer macro for self-recognition. If a document or the global template contains this macro, Mentes considers it to be already infected and leaves it alone.

The virus avoids problems that can occur when 'saving as' a template. Under *Word 6* a new document can only be saved as such in the Template directory. To overcome this, the virus creates a new document (based on the old, infected document, not on NORMAL.DOT), then displays the standard FileSaveAs dialog to obtain the new name and location from the user, saves it there, and finally infects the

new document. At this point, both the old and the new documents are open, so Mentes has to close the old one. In order not to trigger the payload (the AutoClose macro), the virus disables automacros for the duration of this close operation. Given that such effort is made to avoid the Save As problem, we must conclude that the virus was developed in *Word 6*, as *Word 7* can save templates to any directory.

Mentes uses a primitive stealth mechanism by replacing the Tools/Macro command with a routine that displays a message box suggesting that this function is not installed. Despite taking that precaution, it does not intercept the Organizer command, so its macros can easily be removed, manually. Interestingly, the virus also has a ListMacros macro. This command appears in the list of the available commands in the Tools/Customize dialog, but it represents a list box that can be placed on the toolbar and lists the currently open macro editing windows. Like several other similar items, it only represents a dialog item, not a built-in command, and thus cannot be executed. Therefore, the ListMacros macro does not intercept any user action, and is pretty much useless.

The virus has a built-in on/off switch. If the MY.INI file is present in the *Windows* directory, and in the section Word Info, the value of the key Kod happens to be 'aaa' (unlikely to be encountered anywhere but the virus writer's PC), the virus will not infect the global template. If the template is already infected, Mentes will not infect any further documents. Moreover, upon opening the next file, the virus will try to remove itself. It is not quite clear why Mentes does not remove all of its macros. The Killer and FileOpen macros remain in NORMAL.DOT while AutoOpen and Killer are left in documents. Possibly this is supposed to serve as a mechanism to clean any infected documents that are opened subsequently, but due to a bug it fails, producing an error dialog. Nevertheless, the document is converted to a template which can pose problems familiar to those who have suffered such viruses.

Were MY.INI to be removed again, it may be expected that the virus would live on with its two remaining macros. Due to another bug, this is not so: the Organizer.Copy command in the infection routine fails on every FileOpen, resulting in more error messages.

Payload

The most interesting thing about this virus is its payload. Mentes literally steals the contents of documents as they are closed, collecting them for future use. *Nota bene*, this mechanism could have its uses in a backup utility macro. When a document is closed, the virus opens the file C:\LOGIN.SYS (or rather, renames it LOGIN.TXT, opens that and after the write operation is complete renames it to LOGIN.SYS again). It appends to this file the document name, date and time of closing, the first 65,281 characters of unformatted text, and the word docvége ('end of document'). This file will contain the (partial) contents of all documents closed since the last successful network upload.

Mentes then attempts to access the F: drive and connect to \\HS_WORKH\COMMON\STUDENT\TEMP. If these attempts fail, the virus aborts the upload action. Otherwise the virus searches, by a trial-and-error method, for a drive it can write to. Starting at drive D: (and consecutively through to Z:), Mentes attempts to create and remove a directory named Q. If the first writeable drive after C: is the specified network drive, Mentes then uploads the LOGIN.SYS file (the payload was clearly designed to work in a particular LAN configuration, perhaps in the virus writer's school or university computer lab). The contents of each closed document are uploaded to separate files.

The uploaded files are placed in a file ring buffer made of files named Archive.a10, Archive.a11 consecutively to Archive.a50. The current extension counter is stored in the PROG.INI file in the root directory of the network drive. It is increased whenever a new file is uploaded and when it reaches 51, is reset to 10 and the file at the beginning of the ring buffer is overwritten. The files wait on the network drive for someone (presumably the virus writer) to pick them up. If the computer is not connected to a network, an untrappable error message is displayed.

An unfortunate side-effect of this payload is that if the specified network drive is inaccessible (which can be considered to be usual), the size of LOGIN.SYS will increase indefinitely, containing the beginning of each document closed since the initial infection.

Conclusion

Since Mentes has been found in several places in the wild, it must be considered to be more than a theoretical threat. Also, its payload shows a rare type of targeted data theft which shows that the damage viruses cause is not limited to destroying data but extends to exposing sensitive information to unauthorized eyes. Although it is possible Mentes was originally written for legitimate purposes, its disk space-wasting payload and spread far beyond its 'home environment' are testament to the consequences of trying to harness viral code to 'good' ends.

Mentes	
Aliases:	WM/Mentes.
Type:	Word 6/7 document infector with limited stealth.
Self-recognition:	Files containing a macro called Killer are presumed already infected.
Payload:	Saves 65,281 bytes of text from host document to a local file then attempts to upload it to a specific network share.
Disinfection:	In a clean Word environment, delete viral macros from infected document and template files via Organizer.

VIRUS ANALYSIS 3

Brewing Up with the CAFEBABE

Costin Raiu,
GeCAD, Romania

Do you remember the good old days, when an anti-virus researcher only had to deal with conventional file and boot viruses? Maybe not, but I do. In the six years I have worked in this field, the virus scene has changed dramatically, with viruses continually taking on new forms.

We saw multi-partite viruses, then polymorphic boot viruses. Batch infectors appeared, 'inserting' polymorphic viruses designed to slow scanners, followed by NE and PE (*Windows 95* and *NT*) viruses. Then came macro viruses, targeting most versions of *Word*, *Excel*, and recently *Access*. For each new virus type the anti-virus industry has had to 'adapt' to the new conditions and invest huge amounts of resources (time and money) into researching new engines, new file formats and so on.

Having watched the steps taken by the virus writers in the past, I thought that there was little more to surprise me. However, a new sample sent to me by a fellow anti-virus researcher was indeed a surprise – a Java virus.

But Isn't Java Virus-proof?

Java viruses have long been a hot topic. Questions such as 'Is it possible to write a Java virus?' or 'Could a Java virus spread from computer to computer, maybe via the Internet?' have generated a lot of traffic on many discussion lists and newsgroups. The main argument against Java viruses is that applets are run in a highly controlled environment, called the 'sandbox'. An applet, as mentioned above, is a Java program designed to be run in web browsers, but without having access to files or arbitrary network connections on the Java computer.

However, Java also allows you to build real applications, which have full control (in the running context) over the system, like any standard program. Real Java applications cannot be run by web browsers such as *Netscape Navigator* or *Microsoft's Internet Explorer* (*IE*). Therefore, a Java virus could (theoretically) only work as an application, and not as an applet.

Of course, if the sandbox is not implemented correctly, a malicious program could (again, theoretically) 'escape' from its cage and gain access to the various resources provided by the Java Virtual Machine (JVM). Fortunately, the current versions of both *Netscape Navigator* and *IE* have no known JVM implementation problems of the sort necesary to allow this.

Thus, for a Java program to replicate (requiring access to files on the local machine), it must run as a full Java application and not an applet. As Java applications are relatively rare compared to Java applets (which can be found on many web pages), the chance of 'in the wild' infections seems low.

A Strange Brew Indeed

The sample of Java/StrangeBrew I received was around 4 KB in size. It was able to infect other class files and the infected files could infect further, so it is really a virus. StrangeBrew is a native Java virus, which is able to infect both applets and applications. However, it can only spread if run as an application, using the JAVA.EXE program from the JDK (and equivalents on other operating systems), or a similar tool such as the *JView* utility from *Microsoft*.

It will not spread if launched from web browsers such as *Navigator* or *IE*. However, it does work if run as a signed applet from *Sun's HotJava* browser, or a browser running the security plugin that allows signed applets to run as full Java applications. The infection will break the applet signature, but a signed dropper is also a possibility.

The virus uses the 'System.getProperty' method to obtain the current working directory (user.dir) then instantiates a 'File' object to list all the files in that directory. It checks each object and, if accessible, whether the size is a multiple of 101 bytes and if the file name ends with '.class'. This is a self-detection test – StrangeBrew assumes such files are already infected. Interestingly, the size test is the same as that used by Win95/Marburg and several other viruses from the Spanish group responsible for it. There is currently no evidence linking StrangeBrew with that group.

StrangeBrew first looks in the current directory for .class files whose size is divisible by 101. When such a file is found, the virus creates a new RandomAccessFile object to access it. The author chose RandomAccessFile instead of DataInput and DataInputStream because it uses 'seek' operations to work with a file – operations that are only supported by the RandomAccessFile object. A class loader could be written without using seek operations, but it was probably much easier to write the parser using them.

The candidate file is opened in read-only mode. Initially the virus only performs some tests on the file – the actual infection routine is called later. One might wonder why StrangeBrew needs to search for infected files in the current directory. The answer is simple – it must load its code from somewhere, because it cannot access its own code from memory. Therefore, it has to look for an infected file, then open it, parse the class data and headers, and load the virus body into two dynamically allocated arrays (2860 and 1030 bytes long, respectively).

The Loader

The routine responsible for loading the virus code into memory is quite complex. It parses the class file directly, using the methods provided by the RandomAccessFile class. After opening a .class file, the virus skips the first eight bytes (the four-byte CAFEBABEh signature and the four-byte version information header). Then it reads the constant_pool_count variable from the header, moving the current read pointer to the constant pool array.

According to the JVM documentation 'the constant pool array is a table of variable-length structures representing various string constants, class names, field names, and other constants that are referred to within the ClassFile structure and its substructures.' Each entry in the constant pool contains a tag byte and a variable amount of data depending on the tag info. The tag byte can have eleven different values, thus to parse the constant pool an application needs to handle each of these tags. StrangeBrew has this ability.

After reading the constant pool, the next six bytes in the header are skipped (the access_flags, this_class and super_class items). It then reads the interfaces_count and skips the array holding interface information (each interface info structure is two bytes long).

Next, the virus reads the fields_count number and skips the fields table. Then it seeks to the offset of the first method in the class, and checks its code size. If the size of the method's code is not 2826 bytes, the virus moves on to processing the next file in the directory. Otherwise, it decides that the file is infected. This is a safe check, because infected files have, as their first method, 'public void Strange_Brew_Virus()', which is the virus' bytecode body.

After finding a copy of itself in a .class file, the virus again reads the methods_count from the header and 2860 bytes from that offset. The extra space (2860-282) is reserved for properties of the class. The virus code is loaded in one of the two dynamically allocated arrays.

The other array is filled with the last 1030 bytes of the constant pool. (The virus has its own entries in the constant pool, which are stored in the last 1030 bytes.) After loading the two arrays with data, a flag is set 'true'. If that flag is still unset after processing all the files in the directory, execution stops as the virus was unable to load its code from a file.

The Infection Mechanism

The infection code is much more complicated than the loader, having around 1000 Java bytecode source lines. As mentioned above, the virus will only reach the infection code if it is able to load a copy of itself from an infected file in the current directory. If that condition is accomplished, it looks in the current work directory for .class files whose size is not divisible by 101. If such a file is found, a new RandomAccessFile object is instantiated, and used to open the host in read-write mode.

Once again the virus skips the first eight bytes of the header and reads the constant_pool_count value. This is stored for later use when the virus adds 123 new entries to the constant pool. Changing the constant pool size and adding new entries is necessary in order to add new bytecode to the class – at the time of writing, I can see no way of infecting a class file without somehow changing the constant pool.

Returning to the StrangeBrew virus, we should mention that the routine used to parse the constant pool is very similar to that used in the virus loader. Thus, the virus contains a great deal of redundant code. The relevant parser code could have been written as a Java procedure (method), but loading it along with the main virus code would be very complicated.

After parsing the constant pool again, the virus saves a pointer to the access_flags member of the class. Then it reads the this_class index in the constant pool, and saves it for future reference. After skipping the interfaces section and the fields section, StrangeBrew saves a pointer to the methods_count and reads the number of methods in another temporary variable. Next, it reads the access_flags property for the first method in the class, and the length of the attribute used to store the method code. After skipping some irrelevant data, the virus loads the code_length property of the method attribute data (whose size is tested for 2826 bytes in the loader) one more time.

The next step is to read all the data from the first method in memory, and create a new header for it. Then it writes the new header and the entire code from the class which was loaded before creating the new header. After that, it reads again the just-written data and stores it in an internal array (it will write it to the file later). Then, the virus writes its native Java bytecode into the new file, and also appends the data saved in the previous test, including the initial code found in the class.

To work correctly, all the code belonging to the virus needs to be parsed dynamically in order to update all constant pool references. This very short, yet powerful, routine is designed to handle all bytecode cases, and it is probably the way the core of a Java virus detection engine should be implemented. As the Java bytecode contains variable references, a simple CRC on the code buffer cannot be used. Therefore, a Java bytecode parser is required to extract the bytecodes, and to CRC them after that.

StrangeBrew's constant pool entries (the 1030 byte array, filled by the loader earlier) are inserted at the end of the host's constant pool. All the entries in this section of the constant pool are then processed to correct for any code relocation that might be necessary. Similarly, some parts of the method information data structure are also patched. Finally, the constant pool count entry in the header is set to match the size change caused by the additional 1030 bytes. The actual routines that work with the class code are quite complex, and a detailed analysis of each piece of code is difficult. The following is just a brief explanation of how this part of the virus works.

As mentioned above, in order to gain control, the virus will rewrite the first method in the class to include a call to itself – the Strange_Brew_Virus() method. During infection the method is padded with NOPs to align the virus code such that the file size will be a multiple of 101 bytes.

Nevertheless, the infection code is buggy. It fails to process the virus body correctly, so infecting some Java class files will result in an 'intended' virus. No error message appears when executing such damaged replicants due to exception handling. Despite this, the infection routine worked well with several small class files. I had no trouble replicating it to custom 4 KB bait classes, and the resulting files were able to carry the infection further.

This, and the reasons pointed out above, means that it is very unlikely this virus will become a serious concern in the wild. However, those high-end anti-virus products that cannot afford to miss such a virus will have to implement Java class loaders and bytecode parsers in their engines (if they have not already done so).

Epilogue

StrangeBrew is the first Java virus. It infects Java class files, but only runs if the file is executed as a native Java application, and not as an applet. It does not work under 'vanilla' *Navigator* or *IE* browsers and was probably written as a 'proof of concept'. Its infection mechanism is both primitive (only searching for target files in the default directory) and quite advanced (in its infection routines).

It should not be very complicated to write encrypted Java viruses and, therefore, polymorphic ones. Detecting them might pose some problems to the anti-virus world, but since Java applications are not actively exchanged, it seems unlikely these will be seen in the wild. This parallels the *Access* virus situation, but the technical and programming skills required to write a Java infector are much greater.

StrangeBrew

Aliases: Java/StrangeBrew.A.

Type: Non-resident, direct action Java class file infector.

Self recognition: Files whose size is exactly divisible by 101 are assumed infected. It locates its bytecode in such files by checking the size of the first method is 2826 bytes.

Hex pattern:

```
3626 1506 1008 0715 2615 2564
0460 6860 6036 06A7 0066 0615
0604 6415 1610 1860
```

Payload: None.

Disinfection: Delete infected files and recompile, or replace from originals or backups.

COMPARATIVE REVIEW

Half Full or Half NT?

Starting a *VB* comparative of this size is a sobering prospect, much like beginning Hercules' labours with a tight deadline attached. Three thousand boot sector tests and some three-quarters of a million file tests later, the results are out and begging for analysis.

NT is now a well-established and growing platform, with more advanced versions still a distant prospect. Therefore, it should be expected that the products reviewed were able to take advantage of this stable background, detecting well, and with the minimum of glitches.

As ever this turned out not to be the case, and it was not just the new versions causing aggravation or frustration with their ability to lock the test machines. Who were the dismal failures hanging their heads in shame, and who the virus-vanquishing heroes? Read on.

Test Procedures

The platform used for these tests was *NT 4.0* with service pack 3. The same machine was used for all time-tests, while two other hardware-identical machines were employed in conjunction for the scanning processes.

In all cases, the software was deployed in its standard configuration, unless this removed such useful features as on-access scanning, and was run from the Administrator usercode. Several products were submitted along with pleas from their developers that default settings not be used, since they did not scan, for example, MDB files, and that 'all files' be used as an option. For fairness' sake, all such pleas were ignored, as several products which would also detect such viruses with their settings changed were not accompanied by similar requests.

The June WildList was used as the basis of the In the Wild test-set. This, in conjunction with the ever-expanding Macro, Polymorphic and Standard *VB* test-sets, was tested against products submitted by the 3 July deadline. Of special note was the addition of Win95/Marburg and four Win95/CIH variants to the set, which is discussed later.

Also of note were the first VxDs to grace the *VB* test-set in the form of Navrhar. Another interesting 'new addition' was WM/Pwd.A, a macro virus which password-protects infected files. Several products were unable to open the files, which was counted as a non-detection, compared to those which were adamant that a virus was present.

Scan tests were run where possible from CD, thus removing the need to restore files after each scan as a precautionary measure against over-keen deletion or disinfection. Several products, however, generated report files that were either useless or nonexistent. In these cases deletion or quarantining were used in order to produce meaningful results.

Timing tests were run on various operations. On-access scanning overhead was tested using XCOPY to move large numbers of executables, the results being compared against a baseline and normalized across the products. Floppy disk speed tests were performed upon two almost identical disks, differing only in that the files on one were universally infected with Natas.4744.

The hard disk scanning test, combining speed and false positive testing on 5500 executables in the *VB* Clean test-set, should produce results directly comparable with results in the last *NT* review.

The complete detection tests are reported in the main tables. The results reported in the summaries are only the on-demand ones, plus the on-access result for the combined In the Wild test-sets, where applicable.

Alwil AVAST32 v7.70

ItW Overall	100.0%	Macro	98.7%
ItW Overall (o/a)	n/t	Polymorphic	94.8%
ItW Boot	100.0%	Standard	98.4%

Commencing with a sound VB 100%-worthy result, *Alwil's* product continues to put in good performances. All cannot, however, be said to be rosy. *AVAST32* is the second slowest of the products tested when faced with the Clean test-set – in the region of half the scanning rate of the next fastest product. This set also caused *AVAST32* to throw up some cryptic error messages, which declared that the files involved were untested due to 'error e100 f125'.

As the first-encountered product in this review, *AVAST32* also sets the precedent of missing A97M/AccessiV, Win95/Marburg, Navrhar and Win95/CIH. Since CIH and Marburg are flavour of the month, these are discussed later in some detail.

In terms of ease of interface use, *Alwil* has done enough to be rated above average, with no tasks causing particular difficulty. The same cannot be said of the on-access detection routines, however. Although present, these are only able to detect viruses upon execution. Since executing samples, rebooting and rebuilding the machine from disk image backups some 17,000 times is a little impractical, the on-access scanner was left untested against the file viruses. On-access detection of ItW Boot set showed the age-old problem of non-detection if faced with boot sectors with 'strange' BPBs. This has been discussed at great length in *VB's* two preceding *NT* comparatives.

On-demand tests	ItW Boot		ItW File		ItW Overall	Macro		Polymorphic		Standard	
	Number	%	Number	%	%	Number	%	Number	%	Number	%
Alwil Avast32	88	100.0%	665	100.0%	100.0%	1490	98.7%	13500	94.8%	952	98.4%
CA Cheyenne Inoculan	88	100.0%	665	100.0%	100.0%	1338	90.3%	13489	93.8%	952	98.4%
Command AntiVirus	88	100.0%	665	100.0%	100.0%	1498	99.2%	13494	93.9%	952	98.4%
Cybec Vet AntiVirus	88	100.0%	665	100.0%	100.0%	1441	97.4%	13500	94.8%	947	97.9%
Data Fellows FSAV	88	100.0%	665	100.0%	100.0%	1501	99.5%	14244	100.0%	1006	99.7%
DialogueScience Dr Web	87	98.9%	665	100.0%	99.9%	1465	98.9%	14244	100.0%	1006	99.7%
Dr Solomon's AVTK	87	98.9%	665	100.0%	99.9%	1461	98.7%	13500	94.8%	961	98.7%
EliaShim ViruSafe	87	98.9%	659	99.6%	99.5%	1360	92.9%	13243	91.7%	946	97.9%
ESET NOD32	88	100.0%	665	100.0%	100.0%	1461	98.5%	13813	96.4%	970	98.8%
GeCAD RAV	88	100.0%	656	99.2%	99.3%	1480	99.0%	13483	92.1%	901	93.7%
Grisoft AVG	73	83.0%	663	99.7%	97.7%	1243	83.8%	12996	90.4%	936	97.1%
H+BEDV AntiVirNT	86	97.7%	602	94.6%	95.0%	1419	95.9%	10959	76.1%	940	95.9%
Kaspersky Lab AVP	88	100.0%	665	100.0%	100.0%	1501	99.5%	14244	100.0%	1015	100.0%
NAI NetShield NT	88	100.0%	656	99.4%	99.5%	1446	97.7%	13435	92.7%	945	97.9%
Norman TBAV	88	100.0%	665	100.0%	100.0%	1447	97.7%	13496	93.0%	981	98.9%
Norman Virus Control	88	100.0%	665	100.0%	100.0%	1435	96.8%	13498	93.9%	973	97.1%
Proland Protector Plus	25	28.4%	307	49.8%	47.3%	589	39.1%	1465	9.5%	257	36.1%
Sophos SWEEP	88	100.0%	665	100.0%	100.0%	1454	98.2%	13810	96.4%	959	98.3%
Symantec Norton AntiVirus	88	100.0%	665	100.0%	100.0%	1417	95.8%	13500	94.8%	952	98.4%

CA Cheyenne Inoculan v4.00

ItW Overall	100.0%	Macro	90.3%
ItW Overall (o/a)	94.9%	Polymorphic	93.8%
ItW Boot	100.0%	Standard	98.4%

While the general trend in products reviewed seems to be of gradual improvement, *Computer Associates* (*CA*) has seen fit to continue flying in the face of fashion. Scanning the Clean test-set, *Inoculan* continued its unenviable record of causing access violations, crashing *NT* when faced with the unarchiving utility unp.exe. Not overly fast when this program was removed, floppy disk scan speeds were also somewhat greater than the mean, while overhead for the resident portion of the program was average. Having said that, the 'average' overhead seen in these tests was something in the order of 100% – doubling the time taken to copy files, and most certainly a painful side effect.

On-demand scanning proved uncharacteristically quick and easy for boot disks, yet astonishingly slow for the file viruses. Log files were impossible to obtain, since printing results to a file resulted in lines garbled by *Inoculan's* cunning use of linefeeds and pagebreaks.

On-access, the boot sector scanner sent *NT* into blue-screened apoplexy on several occasions – it mattered little whether the disk proffered was infected or not. The on-access scanner was still unable to penetrate the mystery of strange boot sectors.

Despite this, *CA's* product managed to gain a VB 100% award and reasonable, if not notable, detection in other areas. To carry on with this tester's metaphor – it is likely that the *Inoculan* user will compare it to the shirt of Nessus, as worn by Hercules in his later days. It certainly offers some degree of protection but the agony involved in using it is out of all proportion to its utility.

On-access tests	ItW Boot		ItW File		ItW Overall	Macro		Polymorphic		Standard	
	Number	%	Number	%	%	Number	%	Number	%	Number	%
CA Cheyenne Inoculan	75	85.2%	664	99.9%	94.9%	1338	90.3%	13489	93.8%	952	98.4%
Command AntiVirus	75	85.2%	665	100.0%	94.9%	1432	97.6%	13494	93.9%	952	98.4%
Cybec Vet AntiVirus	87	98.9%	665	100.0%	99.6%	1433	96.8%	13000	91.3%	944	97.6%
Data Fellows FSAV	88	100.0%	665	100.0%	100.0%	1497	99.2%	14244	100.0%	1006	99.7%
Dr Solomon's AVTK	87	98.9%	665	100.0%	99.6%	1465	98.9%	13500	94.8%	961	98.7%
EliaShim ViruSafe	n/a	n/a	659	99.6%	n/a	1362	93.1%	13243	91.7%	946	97.9%
ESET NOD32	88	100.0%	665	100.0%	100.0%	1461	98.5%	13813	96.4%	970	98.8%
Kaspersky Lab AVP	88	100.0%	665	100.0%	100.0%	1501	99.5%	14244	100.0%	1015	100.0%
NAI NetShield NT	88	100.0%	656	99.4%	99.6%	1449	97.9%	13275	88.3%	970	98.5%
Norman Virus Control	n/a	n/a	665	100.0%	n/a	1435	96.8%	13498	93.9%	973	97.1%
Sophos SWEEP	88	100.0%	665	100.0%	100.0%	1454	98.2%	13748	96.1%	959	98.3%
Symantec Norton AntiVirus	75	85.2%	665	100.0%	94.9%	1421	96.0%	13500	94.8%	948	98.1%

Command AntiVirus v4.51

ItW Overall	100.0%	Macro	99.2%
ItW Overall (o/a)	94.9%	Polymorphic	93.9%
ItW Boot	100.0%	Standard	98.4%

The third VB 100% award in a row – what is the world coming to? Proof that improvement is possible comes in the all-new incarnation of *F-PROT*. Although a new version of the product, there were no stability problems to be seen with *Command's* packaging of the *F-PROT* engine. Somewhat surprisingly, given *F-PROT's* reputation, macro detection was not 100%. This was partly due to the A97M/AccessiV variants, though to be fair it does not claim to detect these. More surprisingly, it missed the macro portion of Navrhar. The latter is possibly classifiable as a dropper, yet still falls well within the 'should find' category.

A new addition to *Command AntiVirus* (*CSAV*) is an on-access scan for boot sectors, but as yet, strange boot configurations are enough to confound detection and the detection of disk changes is also less than admirable. The on-demand scanning of diskettes is a joy, with the exception of those selfsame strange file systems adding options to the process, which might be considered confusing.

Since its last outing on this platform *CSAV's* polymorphic detection has almost doubled in percentage terms, from 47.6% to 93.9%, and is now back in the realms of the respectable. Some improvement could perhaps be made to the speed, and the on-access overhead is certainly over the desired value. That said, there has been considerable positive feedback on *CSAV's* development since this version was first released. The future may well be promising

Cybec Vet AntiVirus v9.80

ItW Overall	100.0%	Macro	97.4%
ItW Overall (o/a)	99.6%	Polymorphic	94.8%
ItW Boot	100.0%	Standard	97.9%

The rash of perfect on-demand detection against the In the Wild test-sets continues apace with *Vet*. Notorious for its speed, this antipodean offering did not fail to impress on this front. It was third against the Clean test-set, as well as being ahead of average in diskette scanning and least burdensome in the overhead category.

In fact, the top two performers in the overhead category produced one of the more impressive results in this review, in that rather than slowing down XCOPY, the on-access scanners caused the process to become faster. The developers of both these products attribute this unlikely result to their decision to implement on-access scanning as a file-system filter rather than as a service.

Vet performed a little oddly – on a par with *CSAV* in this respect – in that, on-access, it detected all the samples in the Standard test-set, despite failing to do so on-demand. This strangeness was heightened by the reverse being true in some other test-sets and equality prevailing in others.

As far as boot viruses were concerned, inconsistency was noted again, in the missing of ABCD on-access. On-demand, affairs were much happier than in the last test for *Vet*. On that occasion, all non-standard boot sectors were undetected, but this time they were discovered with no problems at all.

Data Fellows F-Secure Anti-Virus v4.01aβ

ItW Overall	100.0%	Macro	99.5%
ItW Overall (o/a)	100.0%	Polymorphic	100.0%
ItW Boot	100.0%	Standard	99.7%

A chimeric breed of *AVP* and *F-PROT*, the *Data Fellows* product has proven unpredictable and bothersome in *Windows 95* reviews, and this trend seems likely to continue. The interbreeding of the two products has certainly given rise to a perceptive beast, though slightly less so than *Kaspersky Lab's* offering, and not without its concomitant problems. As a relatively new product, however, teething problems are to be expected.

Two engines obviously add to the burden imposed upon operations. With on-access scanning enabled, copy operations took four times longer, while other scanning operations were also slow. More disturbing was the logging of infections, which produced double reports for some infected objects, one report for others, and in some uninfected objects resulted in an error message when *AVP* attempted to scan after *F-PROT*.

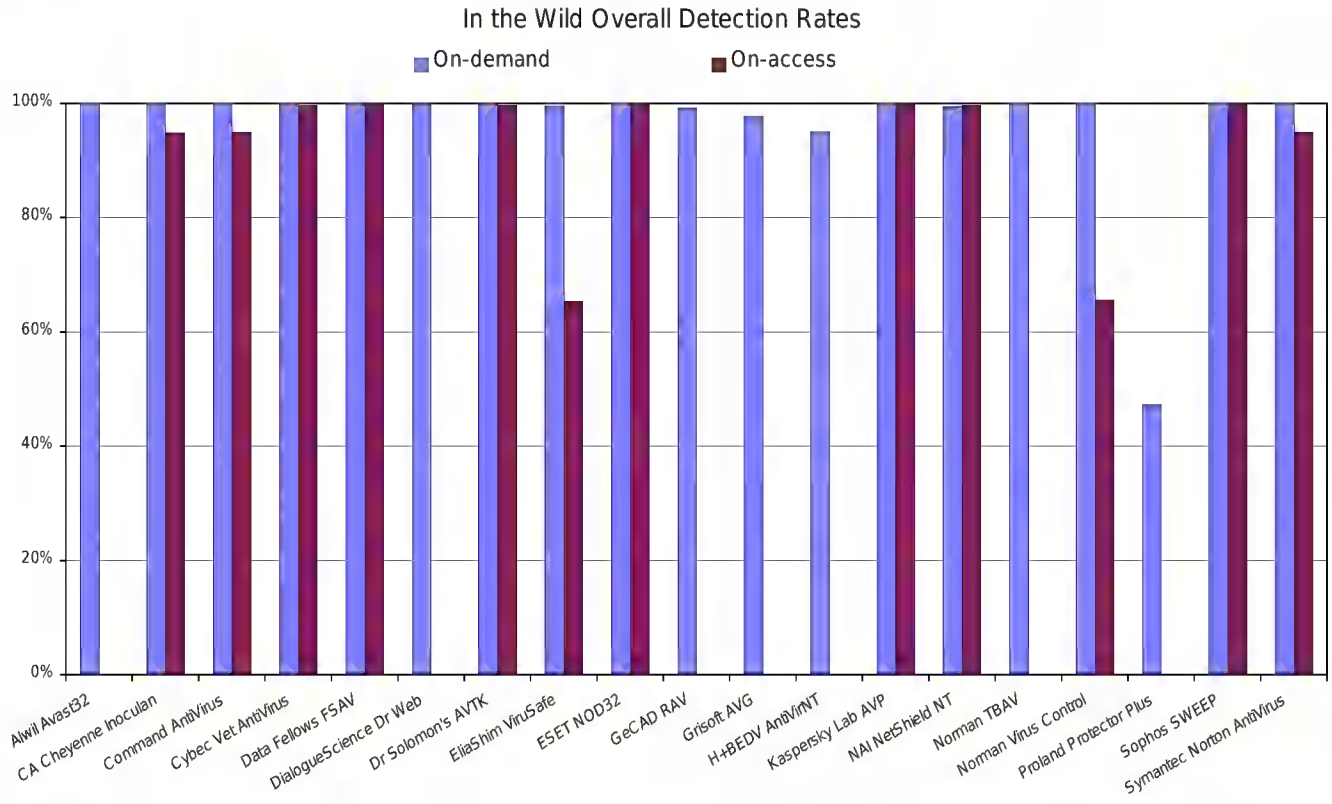
Perhaps due to the Medusa-like ugliness of these reports, *Data Fellows* seems most unwilling to allow log files to be produced, and the tester's tender sensibilities were further shielded by *F-Secure's* (*FSAV*) ability to crash when logs were redirected to a file masquerading as a printer.

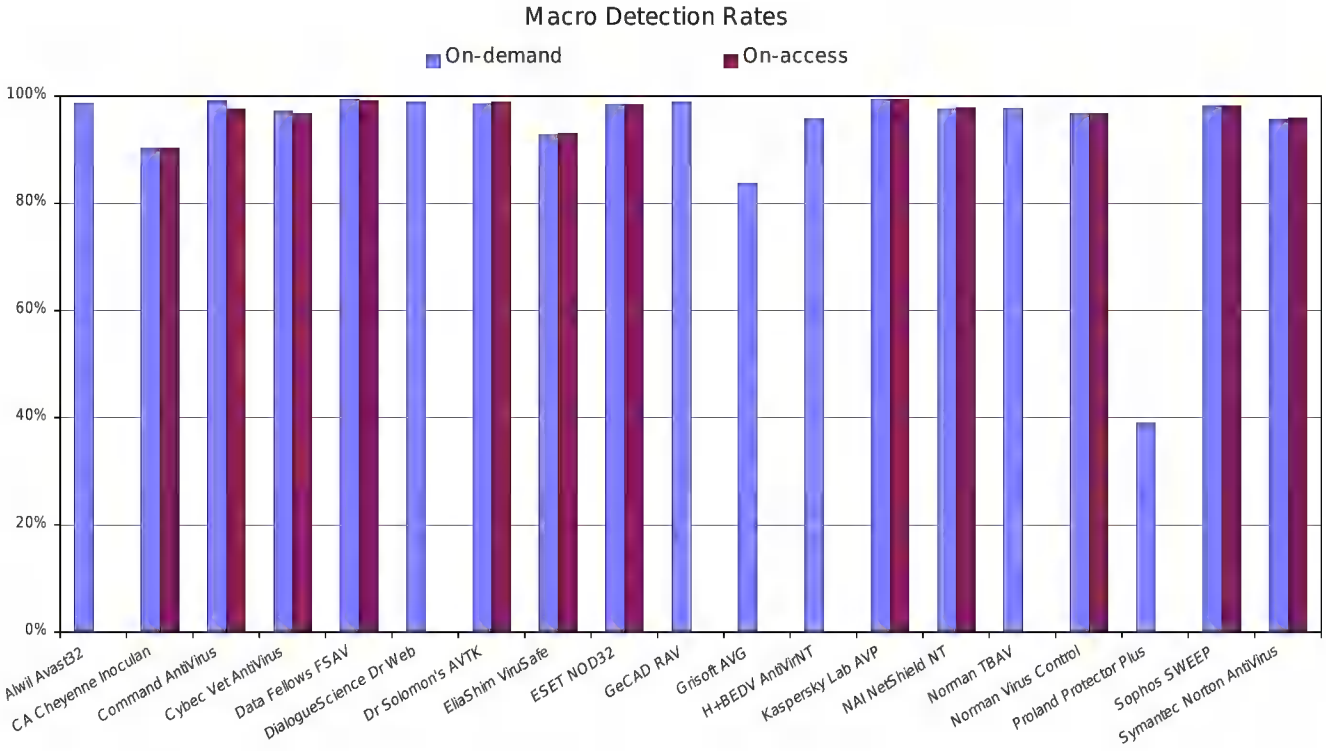
This activity required testing of the maim and kill variety, the program being set up to delete any viral files found, with those remaining taken to be missed samples. Unfortunately, the two 'heads' of the program are often at odds as to whether a sample is viral. The result was a file not deleted but renamed – the first letter of the extension being replaced with V. Not entirely unreasonable it might be thought, until it is realized that Navrhar infects VxDs, files with an extension which tells *FSAV* the file has already been scanned! Thus, although *AVP* can detect the viral VxD, it passes as undetected by the *Data Fellows* product.

Boot sector testing was not exactly a pleasure to see, with the on-access component reducing *NT* to a blue screen on occasion. Problems were also encountered with multiple messages and changeover detection. With a beta version being tested, it is to be hoped that many of these problems have been addressed in the full release. *FSAV* still received a VB100% award despite all these woes.

DialogueScience Dr Web v4.01β

ItW Overall	99.9%	Macro	98.9%
ItW Overall (o/a)	n/a	Polymorphic	100.0%
ItW Boot	98.9%	Standard	99.7%





Dr Web has all the attributes of a mighty club – somewhat slow, a little old-fashioned looking but very effective none-the-less. Heuristics are the order of the day at *Dialogue-Science*, and effective they are indeed. Misses were due mostly to unscanned extensions, though the two W97M/Class variants escaped. The downside of this reliance on heuristics is the announcement of 14 false positives against the Clean test-set, together with the slowest performance in that test – over thirty times longer to perform the scan than the fastest credible scanner.

A VB 100% award eluded *Dr Web* by one missed boot sector virus, Lilith, a non-detection which should be easily rectified. On-demand diskette scanning also proved a little burdensome in that the scan target was reset after each scan had been performed, necessitating individual selection for the 88 disks. A great plus point, from a reviewer and user point of view, was that despite being declared a beta, *Dr Web* showed no signs of instability whatsoever.

A disappointing omission, although admittedly requiring a great deal of programming to remedy, was the lack of an on-access component in this new version.

Dr Solomon's AVTK v7.85

ItW Overall	99.9%	Macro	98.7%
ItW Overall (o/a)	99.6%	Polymorphic	94.8%
ItW Boot	98.9%	Standard	98.7%

The last of a dying breed, the mighty figure that once was *Dr Solomon's* is currently being fitted to *NAI's* Procrustean empire. As *AVTK 8* will never see the light of day, this possibly marks the final outing for the *NT ToolKit* as a fully supported product. As it was, the end came not with a bang but with a whimper, as the missing of Ornate – a virus it has detected in several previous tests – in the boot sector tests denied the product a VB 100% award.

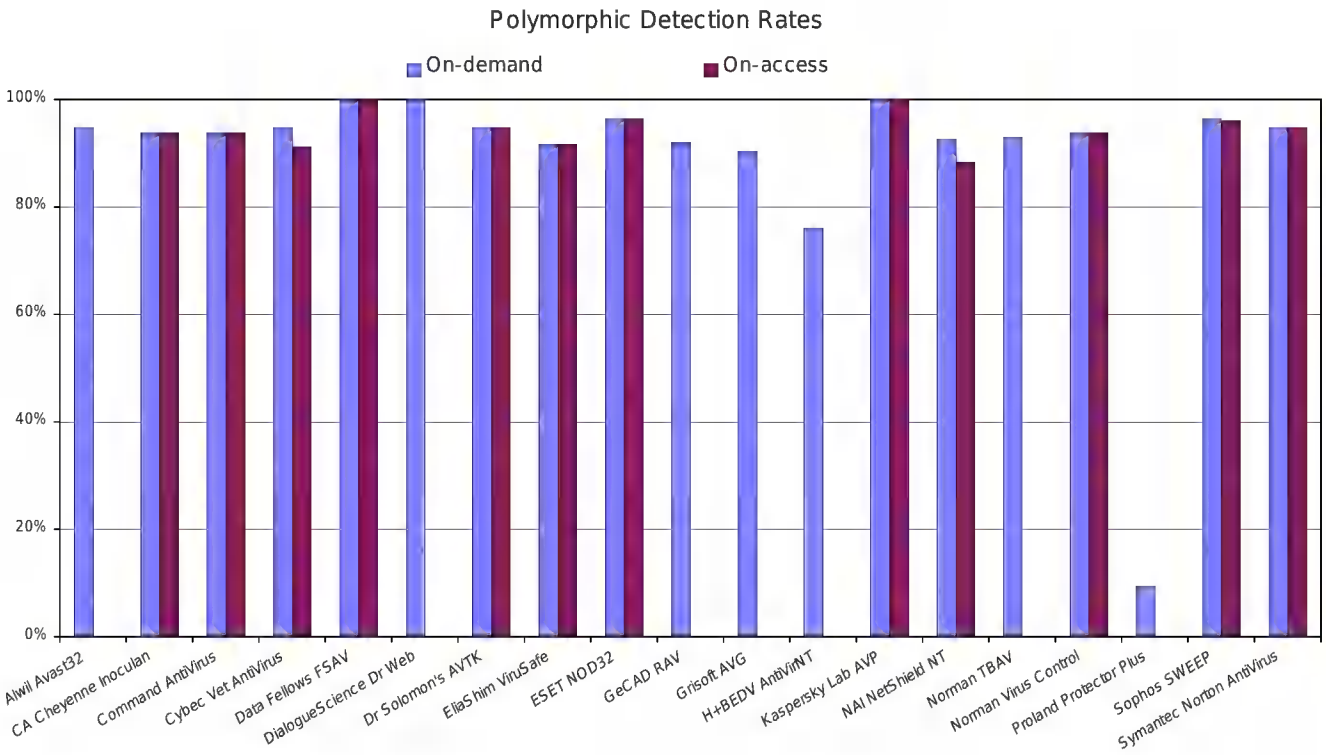
A succession of misses in other areas did more than this, pushing the results well into the mid-range of the detection league. On-access overhead was an area where *AVTK* still remains impressive, not quite up with the best, but only 20% up on times with this component unloaded.

The areas *NAI* hopes to improve on were also behaving at their worst. The selection of subdirectories for scanning proved a Gordian knot in its complexity – *AVTK* twice admitted defeat, with error messages composed entirely of ASCII graphical characters when scans were being prepared. Boot sector scanning was a much more pleasant affair, but the real interest now lies in the alchemical marriage of *Dr Solomon's* and *NAI*.

EliaShim ViruSafe v2.7

ItW Overall	99.5%	Macro	92.9%
ItW Overall (o/a)	n/a	Polymorphic	91.7%
ItW Boot	98.9%	Standard	97.9%

ViruSafe is soon to be enhanced, providing a more complete *NT* product, although the current offering displays no major flaws. Detection was not stunning in any category, though not appalling either – the exception being on-access boot sector scanning which is not supported in any way, shape or form. On-demand detection rates have improved over previous tests, but the perennial favourite Hare.7610 still evades *ViruSafe's* detection routines.



Speed-wise, the hard disk rate continues to be at the very respectable end of the field and now with a much reduced false positive rate, evidence of the continuing development effort. An application which escapes great discussion by doing what it sets out to do and exhibiting no bizarre traits.

ESET NOD32 v1.06

ItW Overall	100.0%	Macro	98.5%
ItW Overall (o/a)	100.0%	Polymorphic	96.4%
ItW Boot	100.0%	Standard	98.8%

ESET has not featured in an *NT* comparative with this dedicated 32-bit product – a situation which often causes trepidation in the reviewer's psyche. The overall, dark cyber-creature theme of the artwork is muted here, but cosmetics are not the prime concern of this review.

The review process was, despite unfamiliarity with the product, a pleasant one overall; the interface being simple to control and effective. On-demand diskette scanning was particularly well-implemented, and with both varieties of boot check there were no problems with either odd boot sectors or disk change detection. Speed was at the better end of the range but on-access overhead was rather high.

Detection, too, was definitely more respectable than many new implementations have managed, earning *NOD32* a VB 100% award on its first appearance on this platform. Results were especially impressive on-access, only lagging slightly behind the *AVP*-powered leaders. *ESET* reports that it is currently busy with translation of its manuals and documentation, and a *VB* standalone review is forthcoming.

GeCAD RAV v6.01

ItW Overall	99.3%	Macro	99.0%
ItW Overall (o/a)	n/a	Polymorphic	92.1%
ItW Boot	100.0%	Standard	93.7%

This version of *RAV* submitted for testing had several notable differences from those seen previously. The addition of some violent colour schemes was quite eye-catching, and the claim to support thirty-four languages marginally more remarkable.

In a more relevant vein, there were also improvements apparent in the internal workings of the program and its detection capabilities. Three incompletely detected viruses against the In the Wild test-set came between it and a VB 100% award. 99.3% overall ItW detection rate is a large and desirable improvement compared to 82.8% in March. Macro detection was second only to *FSAV* for the most improvement, up from 64.3% to 99.3%, and the overall improvements hoist *RAV* firmly toward the top-end of detection performance.

Improvements are still to be had, on the other hand, with nine false positives still raising their heads against the Clean test-set. Lack of an on-access component was none too favourable either, and despite the full detection of boot viruses on-demand, the interface was tortuous at best. Repeated scans required clicking through the selection of a scan, the ignoring of a 'there is something missing' error message, and the choice of various different buttons from a large selection.

Scan speeds are a little sluggish, yet with the current rate of improvement *RAV* is certainly a product to watch.

	Scanning Speed						On-access Overhead (default configuration)	False Positives
	Diskette - Clean		Diskette - Infected		Hard Drive - Clean			
	Time (seconds)	Throughput (KB/s)	Time (seconds)	Throughput (KB/s)	Time (min:sec)	Throughput (KB/s)		
Alwil Avast32	65	15.0	100	11.8	29:16	304.2	n/a	1
CA Cheyenne Inoculan	159	6.1	184	6.4	5:44	1552.7	92.9%	1
Command AntiVirus	124	7.9	133	8.9	3:50	2322.2	123.1%	1
Cybec Vet AntiVirus	61	16.0	66	17.9	1:35	5622.2	-23.9%	1
Data Fellows FSAV	162	6.0	300	3.9	7:53	1129.2	304.1%	0
DialogueScience Dr Web	106	9.2	105	11.3	40:55	217.6	n/a	14
Dr Solomon's AVTK	64	15.2	78	15.2	3:26	2592.8	19.7%	0
EliaShim ViruSafe	59	16.5	65	18.2	2:04	4307.4	93.1%	4
ESET NOD32	57	17.1	65	18.2	2:21	3788.0	154.7%	0
GeCAD RAV	64	15.2	94	12.6	9:43	916.1	n/a	7
Grisoft AVG	63	15.5	71	16.6	2:21	3788.0	n/a	51
H+BEDV AntiVirNT	62	15.7	89	13.3	2:42	3297.0	n/a	4
Kaspersky Lab AVP	61	16.0	67	17.6	5:17	1684.9	172.5%	3
NAI NetShield NT	58	16.8	45	26.3	15:55	559.3	141.8%	0
Norman TBAV	50	19.5	43	27.5	1:12	7418.2	n/a	0
Norman Virus Control	63	15.5	66	17.9	4:26	2007.9	171.6%	0
Proland Protector Plus	62	15.7	85	13.9	1:07	7971.8	n/a	1
Sophos SWEEP	54	18.0	65	18.2	2:25	3683.5	-18.9%	0
Symantec Norton AntiVirus	119	8.2	134	8.8	3:24	2618.2	49.3%	0

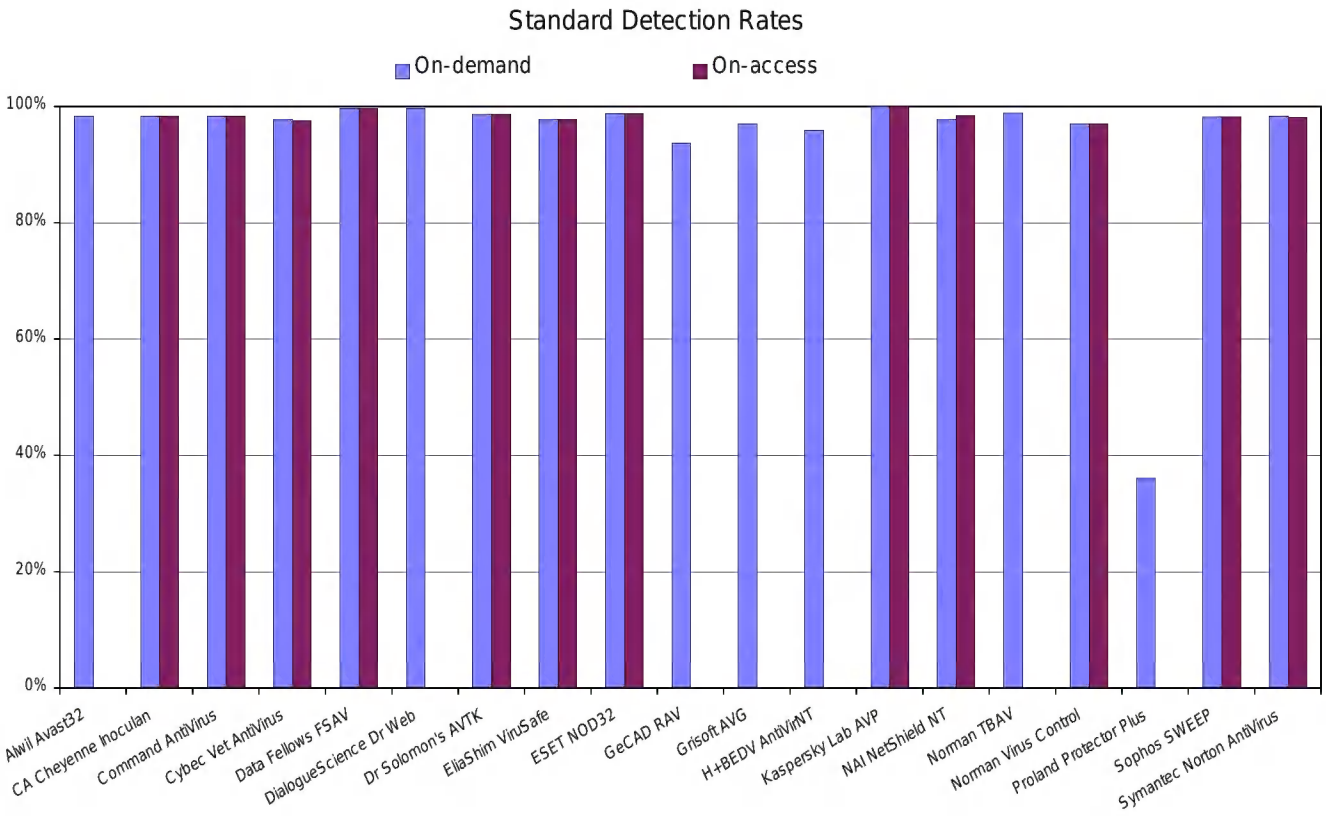
Grisoft AVG v5.0v16

ItW Overall	97.7%	Macro	83.8%
ItW Overall (o/a)	n/a	Polymorphic	90.4%
ItW Boot	83.0%	Standard	97.1%

Shipping as a general-purpose Win32 product, its VxD on-access scanner means *AVG* provides no on-access protection under *NT*. Topping the false positive count with 51 in total – all attributed to the Tentacle virus – *AVG* missed only two samples in the ItW File test. Ironically, these were both samples of Tentacle.10634! This might well be a simple problem with the Tentacle detection string. More problems were apparent in the boot sector tests.

AVG was unable to deal with strange boot sectors in its on-demand tests. It was also unable to detect Hare.7786 and Hare.7610 – both of which have caused problems for many in the past – and that Methuselah of viruses, Natas.4744. In addition to these technical problems, scanning more than one diskette was roundabout and surely off-putting to anyone other than an ardent reviewer.

Grisoft has included some extras not found elsewhere, including a single stepping version of their emulator, and the presentation standards overall are high. Since its first appearance, detection rates have increased but not as remarkably as those of *RAV*. However, *AVG* had the worst boot sector virus detection of the real scanners tested this



issue, finding only 83% of In the Wild Boot viruses. Detection results are less than acceptable in general and it is to be hoped that further redirection of effort towards the internals of the product will reap greater improvements.

H+BEDV AntiVirNT v1.07

ItW Overall	95.0%	Macro	95.9%
ItW Overall (o/a)	n/a	Polymorphic	76.1%
ItW Boot	97.7%	Standard	95.9%

H+BEDV's product provided installation problems, proving to be more paranoid than was healthy for its own good. The first version tested was in English but, upon activation, it failed its integrity check, proclaiming that it was infected and terminating. A newer, post deadline, version was tested and therefore it must be noted that *AntiVir's* results are not directly comparable with other products, reflecting signatures from 30 July.

That said, detection rates were (while not particularly bad in general) certainly under par when it came to the Polymorphic test-set, with a mere 76.1% detection rate. Three of *AntiVir's* four false positives were suspected 'virgen' productions, and correction of this might prove to be a simple tweak. On-demand detection of boot viruses, too, could benefit from some attention, partially due to the product missing samples of Moloch and Lilith and also due to the four keystrokes required for each scan of an infected object. Another offering lacking an on-access scanner, *AntiVir* is looking overdue for a revamp.

Kaspersky Lab AVP v3.0

ItW Overall	100.0%	Macro	99.5%
ItW Overall (o/a)	100.0%	Polymorphic	100.0%
ItW Boot	100.0%	Standard	100.0%

With a very good recent history, it was no great shock when *AVP* qualified for another VB 100% award. However, it was surprising that it missed WM/Mortal.A, but less so that the other missed virus was W97M/Kitty.B – a recent addition to the test-set.

Despite these detection rates, *AVP* was not without some problems. Three false positives and large on-access overheads were not unexpected with the intensive scanning to which *AVP* subjects files. While boot sector scanning was exemplary on-demand, matters were different on-access, the traditional *NT* sticking point. Alerts and change detection were at their seemingly most random, and at one point, perhaps driven to paranoia by detection of too many viruses, *AVP* denied access to the A: drive permanently. This required a reboot to restore things to normality.

Network Associates NetShield NT v3.14a

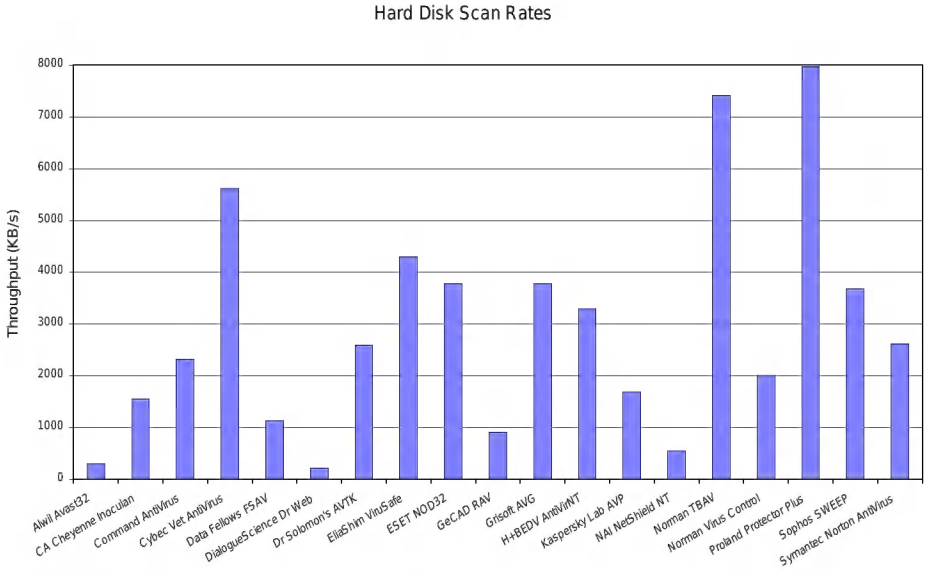
ItW Overall	99.5%	Macro	97.7%
ItW Overall (o/a)	99.6%	Polymorphic	92.7%
ItW Boot	100.0%	Standard	97.9%

Having spent the riches of Croesus on the *Dr Solomon's* engine, this must be the first occasion when *NAI* is hoping to detect fewer viruses than its erstwhile nemesis. These

results do not disappoint on that front. *VirusScan's* next appearance in a *Virus Bulletin* comparative review might well incorporate the *Dr Solomon's* engine, and is an interesting arrival to anticipate.

VirusScan is certainly not as fast as *AVTK*, nor indeed most of the tested products, yet it still managed to produce a seemingly miscounted number of files against the Clean test-set. Overheads were somewhat above average, though floppy disk scan rates were among the best.

An area where *NAI* can claim victory is the boot sector, where detection rates were 100%, something of an improvement upon a notable, if untypical, past performance. The results were good but the interface in the on-demand and on-access versions was still less than perfect. On-demand scanning stopped with the scan start button still depressed, requiring a pause action to allow a new scan even when scanning was clearly complete, while on-access disk change detection and messaging were erratic.



Norman ThunderByte AntiVirus v8.07

ItW Overall	100.0%	Macro	97.7%
ItW Overall (o/a)	n/a	Polymorphic	93.0%
ItW Boot	100.0%	Standard	98.9%

Norman ThunderByte (*NTBAV*) is, as always, vying with *Vet* for the fastest 'real' scanner and on this occasion comes out in front. Speeds on hard disk scanning were more incredible than respectable, more so because it was clear that *NTBAV* was performing a great deal of heuristic analysis. This was visible if the more detailed log file options were selected, when, typically, a half dozen lines of analysis for each virus were produced for the report file.

Floppy disk scanning was in the same speedy league, yet the quickness can only be appreciated if sacrifices are not made. With no false positives, complaints cannot be made on this front, though the lack of an on-access scanner is certainly an oversight. In the grand scheme however, detection rates are the key.

Here again there can be few complaints, since *Norman ThunderByte* is a happy recipient of a VB 100% award. On the negative side, polymorphic detection is worrying at 93% – an area where perhaps speed is causing detection to be cut a little. All in all, *NTBAV* is a virus detector which, unlike many, could afford to become a little more tardy if detection were to increase, and yet again is in need of an on-access component.

Norman Virus Control v4.53

ItW Overall	100.0%	Macro	96.8%
ItW Overall (o/a)	n/a	Polymorphic	93.9%
ItW Boot	100.0%	Standard	97.1%

With much talk of the need for on-access components comes *Norman Virus Control* (*NVC*), which has recently revamped its on-access process. In the past, only macros were protected by the CatsClaw utility but the new version replaces this with a service which scans all file operations but not yet boot sectors. This is only mentioned in passing in the help files for the on-demand scanner, and is otherwise hidden away on the service manager console.

New additions to a product are often prone to hiccups but, thankfully, *NVC* retains its reputation for complete stability. Detection rates were sufficient to gain a VB 100% award, with its main weakness against the Polymorphic test-set. Unusual in this review, results were identical on-access and on-demand, a feature which is linked with the unified service-oriented nature of the *NVC* scanner.

Slight niggles did occur – the on-demand scanning interface is slightly complicated by its need for several clicks, and the overhead for the on-access scanner is rather high.

Proland Protector Plus v6.5

ItW Overall	47.3%	Macro	39.1%
ItW Overall (o/a)	n/a	Polymorphic	9.5%
ItW Boot	28.4%	Standard	36.1%

This is the first appearance of *Proland Software's* product in a Virus Bulletin review. Completely unheard of prior to receipt, it turned out to be very disappointing. References to real scanners earlier in the review may have confused some readers, but it seemed unfair to compare the products discussed to the Augean stable proffered by *Proland*.

Speedy it may be, but the cynical will immediately suggest that the program is doing so little work that anything other than speed would be a miracle.

The virus identities used here seem to have stabilized some two years or so ago – with such wonders of the ancient world as Empire.Monkey.B being too tricky for detection. The figures speak for themselves.

Sophos SWEEP v3.11

ItW Overall	100.0%	Macro	98.2%
ItW Overall (o/a)	100.0%	Polymorphic	96.4%
ItW Boot	100.0%	Standard	98.3%

Alphabetically, *SWEEP* has the dubious honour of following *Protector Plus*, and returns us to the levels of detection expected of a late-nineties anti-virus product. The second of the products to speed up file transfer when used on-access, by some 20%, *SWEEP* is among the faster of the hard and floppy drive scanners too.

The usual worries concerning speed seem to be without foundation in *SWEEP's* case, with a VB 100% award and good detection, though, as with so many of the products this month, polymorphic detection is lower than in the past. Mid infectors continue to make up a good portion of the missed samples in the Standard test-set, though a new version of *SWEEP* in the pipeline offers the possibility that these might in future be detected in a standard scan.

Floppy disk scanning was the fly in the ointment for *SWEEP*, though not for the usual reasons. Interface problems were the key, with the lack of a 'hot' scan-start button and the remarkably small size of the results window being areas where interface design could be improved.

Symantec Norton AntiVirus v4.08

ItW Overall	100.0%	Macro	95.8%
ItW Overall (o/a)	94.9%	Polymorphic	94.8%
ItW Boot	100.0%	Standard	98.4%

Being the last in the line up is an unenviable position for *Symantec's* product, attention compounded by the recent standalone review (see *VB*, August 1998, p 21). Tests against the Clean test-set demonstrated no false positives in an unobtrusively average time, though floppy disk scanning was not as fast as might be hoped. Overheads, on the other hand, were not huge, a matter of great importance to users.

NAV was the final recipient of a VB 100% award, meaning that eleven out of nineteen products qualified for one in this review. Out of the wild and on-access *NAV* looked slightly less convincing than many of the other products, with detection of less than 95% in both the Polymorphic test-set and overall In the Wild on-access.

Conclusion

In summing up, the trend is one of continued good detection against the ItW test-set, with some already noted exceptions. Stability does appear to be a problem with some products, and in the case of *Inoculan* at least, cannot be ascribed to the introduction of new code. In other categories, detection rates are down on past outings, especially in the Polymorphic test-set. The inclusion of new samples – several of them with extensions of SCR, MDB or VXD which are not commonly listed as default file types to scan – contributed here.

The submission date for this review passed shortly before the CIH and Marburg scares were rife, but after the two viruses were known to exist in the field. In light of the subsequent festival of updates and press releases it is interesting to note which products detected these viruses in their submitted versions, if only to mention some of the pitfalls involved with them.

Of the tested software only *DrWeb*, *AVP* and *FSAV* detected all samples of CIH and Marburg which were supplied to them on-demand. The *AVP* engine was aware of the signature patterns involved and acted on them, while *DrWeb* used its heuristic prowess to good effect. *FSAV* includes the *AVP* engine, thus benefitting from *Kaspersky Lab's* speedy inclusion of the virus in its detection library.

Of the more partial detections, *NVC* detected all samples of CIH, whilst *NTBAV* and *NAI's NetShield* detected the majority – missing samples which, although common *Microsoft*-produced files were used as a host, do not strictly follow PE header guidelines. These products are clearly sticklers for the *Microsoft* way to a greater extent than *Microsoft* itself. As for Marburg, both *NOD32* and *SWEEP* were aware of the virus, but neither was able to detect all the samples. Marburg performs several different entry-point modifications depending upon the host file, and neither product seemed to take full account of this.

So, with an extra two months of planning in hand, the next comparative review should, we hope, see most of these problems resolved. On the other hand, that will be the first *Windows 98* comparative, and might include samples of the first Java virus in the test-set – opening a whole new Pandora's box of possible woes.

Technical Details

Test Environment: Three 166 MHz Pentium-MMX workstations with 64 MB of RAM, 4 GB hard disk, CD-ROM drive and a 3.5-inch floppy, running *Windows NT v4.0 (SP3)*. The workstations could be rebuilt from disk images and the master copy of the test-set was held on a CD-ROM. All timed tests were run on one workstation.

Speed and Overhead Test-sets: Clean Hard Disk: 5500 COM and EXE files, occupying 546,932,175 bytes, copied from CD-ROM to hard disk.

Virus Test-set: Complete listings of the test-sets used are at http://www.virusbtn.com/Comparatives/NT/199809/test_sets.html. A complete description of the results calculation protocol is at http://www.virusbtn.com/Comparatives/Win95/199801/protocol.html.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire, OX14 3YP, England

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email: editorial@virusbtn.com
World Wide Web: http://www.virusbtn.com/

US subscriptions only:

Virus Bulletin, 18 Commerce Way, Woburn, MA 01801, USA

Tel (781) 9377768, Fax (781) 9320251

END NOTES AND NEWS

The 25th annual Computer Security Conference takes place at the Chicago Hilton & Towers, Chicago, USA, from 2–4 November 1998. The twelve track conference is preceded and followed by two-day seminars and *Computer Security Institute* members are eligible for a $100 saving off the conference fee. The affiliated exhibition runs from 1–3 November. For more details contact *CSI*; Tel +1 415 356 3371, fax +1 415 905 2218, or visit http://www.gocsi.com/.

An introductory computer virus workshop on 16 September 1998 will be followed on 17 September by an advanced session at the *Sophos* training suite in Abingdon, UK. To register for a place, contact Karen Richardson; Tel +44 1235 544015, fax +44 1235 559935, or find details at http://www.sophos.com/.

To coincide with the *début* of *Notes 5.0*, ***Content Technologies Ltd* is now shipping *MIMEsweeper for Domino*** which provides content-based protection for *Domino* users of email, database and Web. It is available immediately, priced from £1,643 for up to 50 users. For more information contact; Tel +44 118 9301300 or visit the *Content Technologies* web site; http://www.mimesweeper.com/.

***Infosecurity Scotland '98* will be held at Edinburgh's Royal Highland Centre from 28–29 October 1998.** Visitors will be able to see the latest in hacker-proof modems, encryption technology, chip detection products, enterprise-wide network solutions, anti-virus software, anti-theft devices, help desks, business continuity solutions, email protectors and Internet security devices. To register for a free ticket or for more details contact: Tel +44 181 9107790 or visit the exhibition web site; http://www.infosec.co.uk/scotland/.

***Network Associates* (formerly *Dr Solomon's*) is running a live virus workshop from 13–14 October 1998**, priced £695+VAT, at the Barns Hotel, Bedford, UK. For more information, contact Caroline Jordan; Tel +44 1296 318881 or email Caroline.Jordan@drsolomon.com.

Registrations are now being taken for VB'98, to be held at the Munich Park Hilton, Munich, Germany from 22–23 October 1998. Details about the eighth annual *Virus Bulletin* conference and concurrent exhibition can be found at http://www.virusbtn.com/. For further information, or to register for the conference, please contact Jo Peck; Tel +44 1235 555139, or email Joanne.Peck@virusbtn.com.

Compsec '98, the fifteenth World Conference on Computer Security, Audit and Control will take place from **11–13 November 1998, at the Queen Elizabeth II Conference Centre in London, UK.** The agenda includes an exhibition, a pre-conference workshop on 10 November and the Seventh Annual Directors' Briefing on 13 November. Early bird discounts are available for registrations received before 15 May. For details and a registration form, contact the conference secretary Amy Richardson; Tel +44 1865 843643, fax +44 1865 843958, email a.richardson@elsevier.co.uk, or visit the new Compsec '98 web site http://www.elsevier.nl/locate/compsec98/.

The fifth international conference on computer security, audit and control, *COSAC'98* is to take place at the Slieve Donard Hotel, Newcastle, County Down, Northern Ireland from 14–18 September 1998. Features include a pre-conference training day and full partners' programme. For more information about registering for ***COSAC'98*** contact Helen Hawkins; Tel +44 1232 738080 or email cosac@aka-associations.co.uk.

***Trend Micro Inc* has shipped an OPSEC-certified scanner for *Sun's Solaris* operating system.** Intended primarily as an Internet gateway scanner, *InterScan VirusWall 2.6 for Solaris* interoperates with *Check Point's Firewall-1* and other OPSEC-compliant firewalls. More details are available from http://www.antivirus.com/. In further news from *Trend Micro*, on Tuesday 18 August the company debuted on the Nikkei Stock Exchange (Tokyo). In early trading the stock was selling at 93% above its initial public offering of ¥4300. Representatives from *Trend Micro* explained the move as a reflection of several issues. Firstly, the globalization of the anti-virus industry serves as a reminder that the US is not the only software market. Secondly, it is a case-study of whether a Japanese company can use the culture of Silicon Valley to generate success. In light of the Asian financial crisis, this will be an interesting development to watch.

***Network Associates (NAI)* announces the release of a new version of *Gaunlet Firewall for Windows NT*.** Version 2.1 of this product is the first release since the recent merger with *Trusted Information Systems*. Apart from ease of use and performance enhancements, the main changes are architectural modifications preparing the product for integration with other *NAI* security products.